VŠB – TECHNICAL UNIVERSITY OF OSTRAVA

University Study Programmes

Department of Control Systems and Instrumentation

Řízení laboratorního modelu pomocí PLC automatu

Control of the Laboratory Model using PLC Controller

Student: Arunlaxman Palaniappan (PAL0224)

Supervisor: Ing. Miroslav Mahdal, Ph.D.

Ostrava 2019

VŠB - Technical University of Ostrava
University Study Programmes
Department of Control Systems and Instrumentation

Diploma Thesis Assignment

Student:	**Arunlaxman Palaniappan**
Study Programme:	N3943 Mechatronics
Study Branch:	3906T006 Mechatronic Systems
Title:	Control of the Laboratory Model using PLC Controller Řízení laboratorního modelu pomocí PLC automatu
The thesis language:	English

Description:

1. Describe the Siemens PLC - Simatic S7-1500, hardware configuration and the software tools for programming.
2. Describe the existing laboratory model of the thermal system. Connect the model to the PLC and verify the functionality of the model.
3. Identify the laboratory model and perform the synthesis of the control system. Design the PLC control algorithm for control of the laboratory model. Experimentally verify the control system.
4. Study the Simatic WinCC software and design a visualization of the thermal system on the TP700 Comfort HMI Touch Panel.
5. Evaluate the results achieved and the possibilities of their use in teaching or in practice.

References:

PLC MANUAL, 2018. PLC Programming [online]. [cit. 2018-03-19]. Available from: http://www.plcmanual.com/plc-programming.
SYSTEM IDENTIFICATION, 2018. Model Identification - lecture [online]. [cit. 2018-03-19]. Available from: https://web.stanford.edu/class/archive/ee/ee392m/ee392m.1034/Lecture8_ID.pdf.
TECHNICAL DESCRIPTION OF THE SIEMENS SIMATIC PROCESSOR UNIT S7-1516, 2018. S7-1516 Processor unit description [online]. [cit. 2018-03-19]. Available from: http://support.automation.siemens.com/ww/llisapi.dll?func=cslib.csinfo&lang=en&objid=6es75163an000ab0&caller=view.
WINCC, 2018. WINCC software [online]. [cit. 2018-03-19]. Available from:

Extent and terms of a thesis are specified in directions for its elaboration that are opened to the public on the web sites of the faculty.

Supervisor: **Ing. Miroslav Mahdal, Ph.D.**

Date of issue: 21.12.2018
Date of submission: 20.05.2019

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA OSTRAVA UNIVERZITNÍ STUDIJNÍ PROGRAMY

doc. Ing. Renata Wagnerová, Ph.D.
Head of Department

Ing. Zdeňka Chmelíková, Ph.D.
Vice-rectress for Study Affairs

Student's Affidavit

I declare that I have prepared the whole diploma thesis independently under the leadership of the diploma thesis supervisor, and I have stated all the documents and literature used.

In Ostrava on May 20, 2019.

.....
Student's signature

I declare that:

- I am aware that Act No. 121/2000 Coll., Act on copyright, rights related to copyright and amending some laws (the Copyright Act), in particular Section 35 (Use of a work in the civil or religious ceremonies or in official events organized by public authorities, in the context of university performance and use of university work) and Section 60 (university work) shall apply to my final Diploma thesis.
- I understand that VŠB – Technical University of Ostrava (hereinafter referred to as "VŠB-TUO") has the right to use this final Diploma thesis non- commercially for its internal use (Section 35 Subsection 3 of the Copyright Act)
- if requested, a copy of this Diploma thesis will be deposited with the thesis supervisor,
- if VŠB-TUO is interested, I will make a licensing agreement with it permitting to use the thesis within the scope of Section 12 Subsection 4 of the Copyright Act,
- I can only use my thesis, or grant a license to use it with the consent of VŠB- TUO, which is authorized in such a case to demand an appropriate contribution to the costs that were incurred by VŠB-TUO to create the thesis (up to the actual amount),
- I understand that - according to Act No. 111/1998 Coll., on higher education institutions and on changes and amendments to other acts (Higher Education Act), as amended - that this Diploma thesis will be available for public before the defence at the thesis supervisor's workplace, and electronically stored and published after the defence at the Central Library of VŠB-TUO, regardless of the outcome of its defence.

In Ostrava on May 20, 2019.

.....

Signature of the author

Name and surname of the thesis author: Arunlaxman Palaniappan

Address of the thesis author: No:86, Periyar pathai, Choolaimedu,

Chennai– 600094, Tamilnadu, India

ANNOTATION

Arunlaxman Palaniappan. *Control of the Laboratory Model using PLC Controller: Diploma Thesis.* Ostrava: VŠB –Technical University of Ostrava, Faculty of Mechanical Engineering, Department of Control Systems and Instrumentation, 2019, 81 p. Thesis head: Ing. Miroslav Mahdal, Ph.D.

This thesis deals with temperature control which is one of the important tasks in industry. It can be used for industrial applications like steel material processing. Nowadays industrial automation is mainly dependent on PLC. So, PLC based control technique is proposed to work with the temperature control system, where the temperature of the box is controlled by the input voltage of the bulb and fan acts as a disturbance to the system. System step response measurement and identification are done. Later, the step response is converted into transfer function and the controller tuning parameters are calculated. In this thesis, a traditional PID controller is used to achieve the overall control of the system and it is followed by visualization in HMI and data logging process. Both software and hardware design of the system is explained and analyzed in detail.

KEYWORDS

S71500 PLC; Temperature control; PID controller; System identification; Closed loop control system; HMI TP700 comfort Control algorithm; Temperature sensor (TMP36); Fan; Bulb; Data logging.

ANOTACE

Arunlaxman Palaniappan. *Řízení laboratorního modelu pomocí PLC automatu: diplomová práce.* Ostrava: VŠB – Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2019, 81 s. Vedoucí práce: Ing. Miroslav Mahdal, Ph.D.

Tato práce se zabývá regulací teploty, která je jedním z důležitých úkolů v průmyslu. Lze použít pro průmyslové aplikace, jako je zpracování ocelových materiálů. V současné době je průmyslová automatizace závislá především na PLC. Řídicí technika založená na PLC je navržena tak, aby umožňovala řízení teploty, kde je teplota soustavy řízena vstupním napětím žárovky a ventilátor působí jako porucha systému. Bylo provedeno měření přechodových charakteristik a identifikace systému. Později je z přechodové charakteristiky získána přenosová funkce a jsou navrženy parametry regulátoru. V této práci je pro řízení systému použit tradiční PID regulátor a následuje vizualizace na HMI panelu a záznam dat. Podrobně je vysvětlen a analyzován návrh softwarové a hardwarové části systému.

KLÍČOVÁ SLOVA

S71500 PLC; Regulace teploty; PID regulator; Identifikace systému; Řídicí system; HMI TP700 comfort; Řídicí algoritmus; Teplotní čidlo (TMP36); Ventilátor; Žárovka; Záznam dat.

Contents

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AC – Alternating Current

AI – Analog Input

AQ – Analog Output

CFM – Cubic Feet per Minute

CM – Communication Module

CPU – Central Processing Unit

DB – Data Block

DC – Direct current

DI – Digital Input

DP – Decentralized Peripheral

DQ – Digital Output

GUI – Graphical User Interface

HMI – Human Machine Interface

LED – Light Emitting Diode

NIC – Network Interface Controller

OB – Organisation Block

PID – Proportional Integral Derivative

PLC – Programmable Logic Controller

PM – Power Module

PN – Profinet

SCADA – Supervisory Control and Data Acquisition

TIA – Total Integrated Automation

USB – Universal Serial Bus

1 INTRODUCTION

Over the years the demand for high quality, greater efficiency and automated machines have increased in the industrial sector. Modern sensing technology and control methods are undergoing continuous innovation, where the real-time control of the temperature plays an important role in industrial sectors. It demands quick response and higher accuracy now more than ever. Nowadays temperature control is vastly used in all aspects of industrial control process and production techniques.

For example, in the iron manufacturing industry to turn iron and steel into a molten state, it requires a certain temperature and proper heat treatment to achieve this result and produce a perfect quality product. But, achieving the required temperature is a tedious process due to the large inertia and delay with time variable and multi-variable parameters. (Wei, 2016)

Another example is boiler temperature control in power plants, it is one of the most difficult processes to maintain the temperature throughout the system automatically. By making this system automatic we can reduce the human intervention and monitor the process in terms of visualization to reduce error and improve the accuracies. (Gowri shankar, 2008)

At present, the PID control techniques adopt the most complicated cases like temperature control, speed control, etc. PID parameters should be selected according to the system and system response. For the visualization purpose, we can use a supervisory control and data acquisition system or simply known as the SCADA system. We can create a whole plant in a SCADA system to monitor the entire plant and control the important parameter in order to achieve the required process. (Gowri shankar, 2008)

This thesis is mainly focused on controlling the box temperature of the given model. The temperature of the system is controlled by a closed loop control algorithm. This system is designed with disturbance and the disturbance is represented by a fan. The overall goal of this project is to achieve the required temperature even with disturbance acting on the system. PID controller plays an important role in controlling the temperature in the required state. If the controller identifies the disturbance, the output of the controller will produce the required voltage to the bulb to maintain the temperature. Since this project is a small-scale laboratory model, the HMI system is being used instead of a SCADA system. Data logging is introduced to store the values of all tags and later it can be used for analysis. These stored values are used in the future to predict the accuracy and lifetime of the system. In this thesis, I will be explaining about the description of Laboratory model, identification of the system, K_P, T_I, and T_D, control algorithm of a closed loop control system, visualization of the laboratory model and data logging of the system.

2 S7-1500 AND ITS SOFTWARE TOOLS

The SIMATIC S7-1500 is the latest development of the SIMATIC S7 series of automation systems. Through the integration of numerous new performance features, the S7-1500 automation system offers extremely good operability and high performance. Along with this, it also gives us an increased system performance, integrated Motion Control functionality, isochronous real-time input/output and also integrated display for the machine-oriented operation which makes it easy for the user to run diagnostics with ease.

Configuration

The S7-1500 automation system is installed on a mounting rail and can comprise of up to 32 modules. These modules are connected to each other with U connectors.

Fig. 1 Configuration of an S7-1500 automation system

It consists of 4 major parts,

1. System power supply module
2. CPU
3. I/O modules
4. Communication module

2.1 Power module (PM)

The power module consists of system power supply and load current supply, both are explained in the upcoming topics. For the laboratory setup, PM 190W 120/230V AC model was used.

Load current supply (PM)

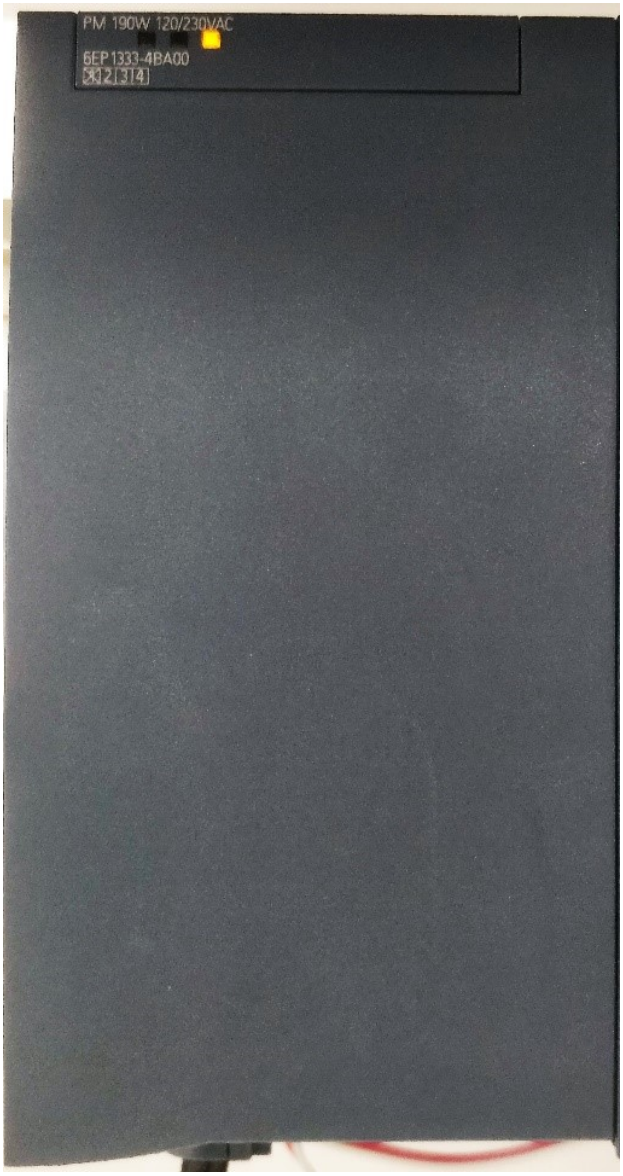


Fig. 2 Load current supply

The central modules (CPU), input and output circuits of the I/O modules are supplied with a 24 V DC supply through the load current supply (PM). The load current supply does not occupy a slot in the configuration and is not included in the system diagnostics. These devices can be mounted on the mounting rail. The load current supply can be configured using STEP 7. Load current supplies are available two models namely, (SIEMENS[12], 2018)

- PM 70W 120/230V AC
- PM 190W 120/230V AC

2.2 CPU

The CPU contains the operating system and executes the user program. The user program is located on the SIMATIC memory card and is processed in the working memory of the CPU. The connection to the process is centralized or distributed via PROFINET or PROFIBUS with I/O modules. The PROFINET interfaces on the CPU allow simultaneous communication with PROFINET devices, PROFINET controllers, HMI devices, programming devices, other controllers and other systems. CPU 1516-3 PN/DP supports operations as an IO controller and I-device. (SIEMENS[12], 2018)

Fig. 3 Central processing unit

Similarly, to the PROFINET interface, the PROFIBUS interface available on the CPU allows communication with other devices. When we use the interface as PROFIBUS DP interface, the CPU on the PROFIBUS DP also assumes the role of a DP master.

The CPU executes the user program and uses the integrated system power supply to supply the electronics of the modules.

Features and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS / PROFINET
- HMI communication
- Integrated web server
- Integrated technology
- Integrated system diagnostics
- Integrated Industrial Security functions

2.3 I/O module

The I/O modules form the interface between the controller and the process. The controller detects the current process state through the connected sensors and actuators and triggers the corresponding reactions. The Equipment stated below are used in the current laboratory setup. S7-1500 PLC consists of a communication model, digital input, digital output, analog input, and analog output. Each part is important for both data storage and communication purpose which is a major part S7-1500 PLC. (SIEMENS[12], 2018) I/O modules are classified into the following types of modules:

- Digital input (DI)
- Digital output (DQ)
- Analog input (AI)
- Analog output (AQ)

Digital input (DI)

Digital input modules contain digital inputs for the automation system. Digital sensors and actuators can be connected to the automation system via these modules. Switches and 2-wire proximity switches can be connected to input modules.

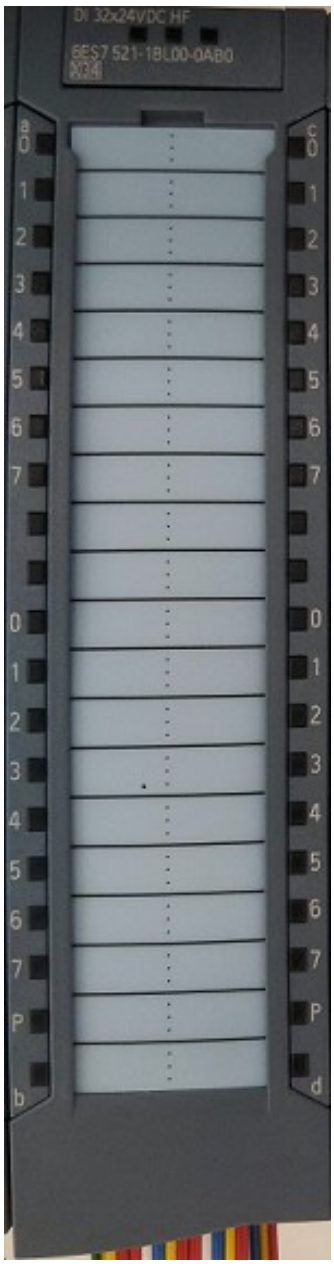


Fig. 4 Digital input (DI 32X24VDC HF)

Properties of DI: (SIEMENS[5], 2018)

It has 32 digital inputs and they are electrically isolated in groups of 16, of which channel 0 and 1 optionally included with counter function. It has an input voltage which is rated at 24 V DC. Its input delay can be configured from anywhere between 0.05 ms and 20 ms. Another important property is that each of its channel can be configured to have its own hardware interrupt. Switches such as 2-/3-/4-wire proximity switches can be connected to the digital input in order to activate the output. This hardware compatible with digital input module DI 16x24VDC HF.

Digital output (DQ)

Digital output modules contain digital outputs for the automation system. Output modules are suitable for the connection of solenoid valves, contractors, small motors, lamps, and motor starters.

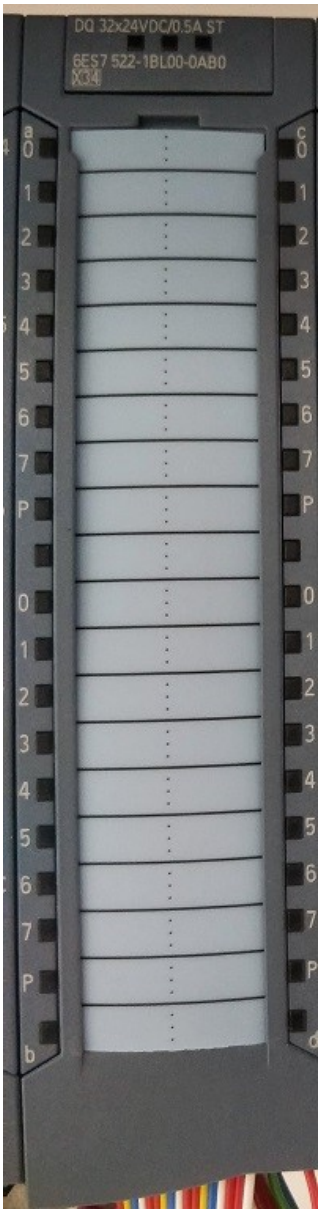


Fig. 5 Digital output module (DQ 32X24VDC/0.5A ST)

Properties (SIEMENS[6], 2018)

It has 32 digital outputs and they are electrically isolated into groups of 8. The maximum output voltage is 24V DC and it has a rated output current of 0.5A per channel. It is suitable for solenoid valves, DC contactors, and indicator lights. This hardware compatible with digital output module DQ 16x24VDC/0.5A ST.

Analog input (AI)

More complex tasks require the processing of analog process signals. Analog actuators and sensors can be connected to a SIMATIC S7-1500 via analog modules without additional amplifiers. Analog input modules facilitate connection of the controller to the analog signals of a process. They are suitable for connecting analog sensors such as voltage and current sensors, thermocouples, resistors, and resistance thermometers as well as analog actuators. (SIEMENS[1], 2018)

Fig. 6 Analog input module (AI 8XU/I/RTD/TC)

Properties

It has 8 analog inputs and each channel can be set to have its own voltage and current measuring type. The suitable measuring type can be selected from a drop-down menu available in channel properties. Resistance thermometers (RTD) is adjustable for channel 0,

2, 4 and 6. Thermocouple (TC) measuring type can be set per channel. The hardware interrupt on limit violations can be set per channel (two low and two high limits per channel).

Analog output (AQ)

Analog Output Module offers solutions used to convert analog values available in factories and plants from the digital format which are generated by the CPU module of the PLC. Analog Output module reference designs can be a channel to channel isolated or group isolated and generates analog values in the range of 0-5V, 0-10V, ±5V, and ±10V for voltages 4-20mA, ±20mA, and 0-20mA for current. (SIEMENS[2], 2018)

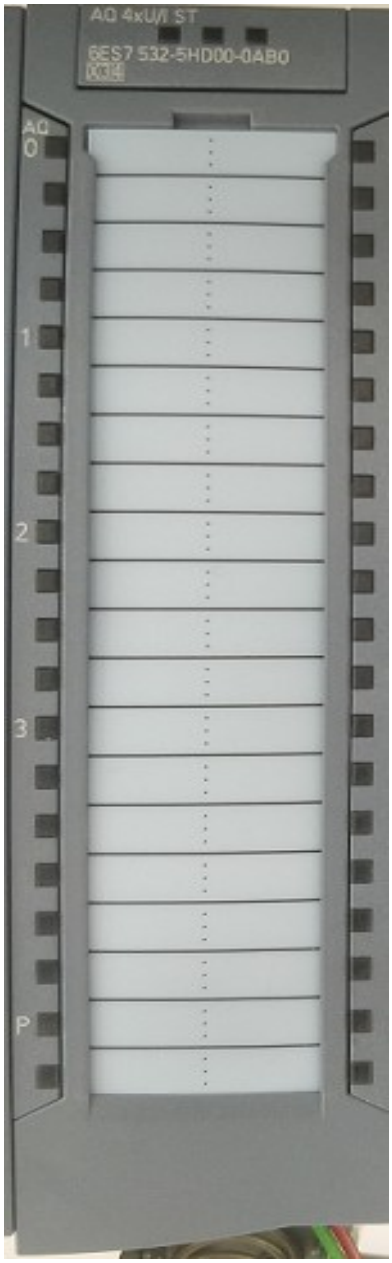


Fig. 7 Analog output module (AQ 4XU/I ST)

Properties

The analog output module has 4 outputs, for current output and voltage output. It has ranges for current output, which are ±20mA with a resolution of 16 bits, 0 to 20mA with a resolution of 15 bits and 4 to 20mA with a resolution of 14 bits. It has ranges for voltage output, which are ±10V with a resolution of 16 bits, ±5V with a resolution 15 bits, 0 to 10V with a resolution of 15 bits and 1 to 5V with a resolution 13 bits. Also, this module Electrically isolated from supply voltage L+.

2.4 Communication module (CM)

The communication module establishes a point-to-point connection which helps in the exchange of data between our PLC and another PLC or a computer. Given below are the properties and applications of the communication module. (SIEMENS[4], 2018)

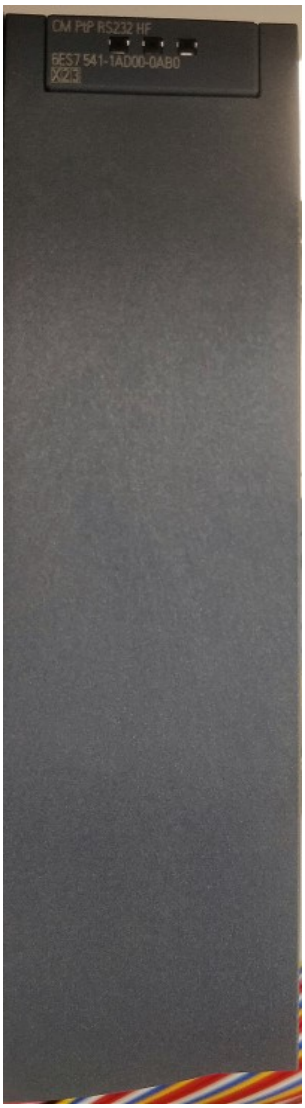


Fig. 8 Communication module (PTP RS232 HF)

Properties

It has an RS232 interface and a data transmission rate between 300 and 115200 bps. It has a frame length of up to 4 kilo bytes and the transmission protocols include Freeport, 3964(R) and Modbus.

Applications:

Central operation and Distributed operations in an S7-1500 system.

2.5 Software Tools

Various types of software tools for the S7-1500 PLC is explained in the upcoming subtopics.

2.5.1 STEP7

STEP 7 is the basic software for configuring and programming SIMATIC control systems. It includes tools and functions for a number of tasks related to automation projects, such as hardware configuration and parameterization, communication definition, programming, testing and project replication, service, document management and archiving, operating and diagnostic functions. Programming can be done in many languages, the basic ones include:

- LD - Ladder Diagram - the language of the contact or relay schemes.
- FBD - Function Block Diagram - function block language.
- SCL – Structure Control Language.
- STL- Structured Text - structured text language.

2.5.2 TIA Portal

TIA Portal is a comprehensive software environment for maintaining data consistency across the entire automation system project, along with a range of powerful library libraries covering all automation features. This is supported by the fact that STEP 7 programs and basic tools for creating visualization applications are now part of the TIA Portal environment. It is also possible to install WinCC directly into the TIA Portal

The SIMATIC STEP 7 in the TIA Portal is the software for configuration, programming, testing and diagnosis of all SIMATIC controllers.

Advantage of working with TIA Portal: (SIEMENS[12], 2018)

1. Full symbolic programming and integrated engineering
    - Reduces the complexity of programming and data management.
    - Process automation and process visualization go "hand-in-hand".
2. Powerful and intelligent editors
    - Five different programming languages are available for implementing your automation task.
    - Considerable time savings in project development using LAD, FBD, STL, SCL, Graph Innovations in the programming languages
    - Single Calculate Box instruction for complex algorithms.
    - Implicit type-conversion for reduced programming.
    - Indirect addressing in all programming languages.
    - Data blocks > 64 kB up to 16 MB.
3. Comprehensive library concept
    - Use the ready-made instructions and pre-existing parts of the project again and again.
4. Motion integration
    - Implementation and data communication of drive axes.
    - Easier programming of axis movements with PLC open blocks.
5. Trace
    - Graphical representation of program variables.
    - I/O signals for efficient real-time diagnosis and optimization.
6. Diagnostics
    - No user programming required for system diagnostics Consistent program download and upload.
    - When downloading, all program changes are loaded automatically.
    - When uploading, all tag names and comments are loaded to an empty PG.
7. Integrated code protection
    - Better protection of intellectual property due to knowhow and copy protection.
    - Extended access protection for CPU.
    - Support Ethernet Security CPs.
8. Comprehensive online/offline comparisons
    - All hardware and software components can be compared online next to offline. This ensures fast recognition of potential differences.

Creating a project

1. Open TIA Portal V15.
2. Click "Create new project".
3. Enter a name for your project.
4. Click "Create" to create a new project.

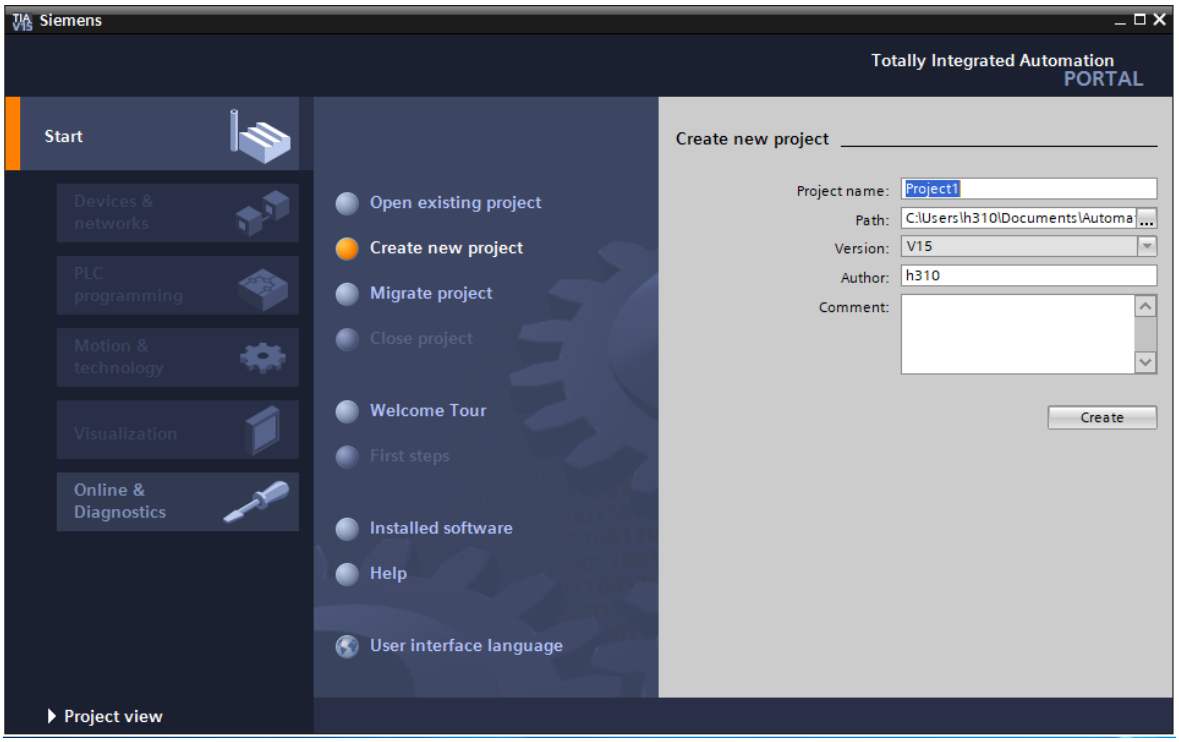


Fig. 9 Create new project window (TIAV15)

Adding an S7-1500 CPU

1. Open the "Devices & Networks".
2. Insert a new device.
3. Open the "SIMATIC S7-1500" folder.
4. Select the CPU which has not yet been specified.
5. Create the CPU with a double-click.

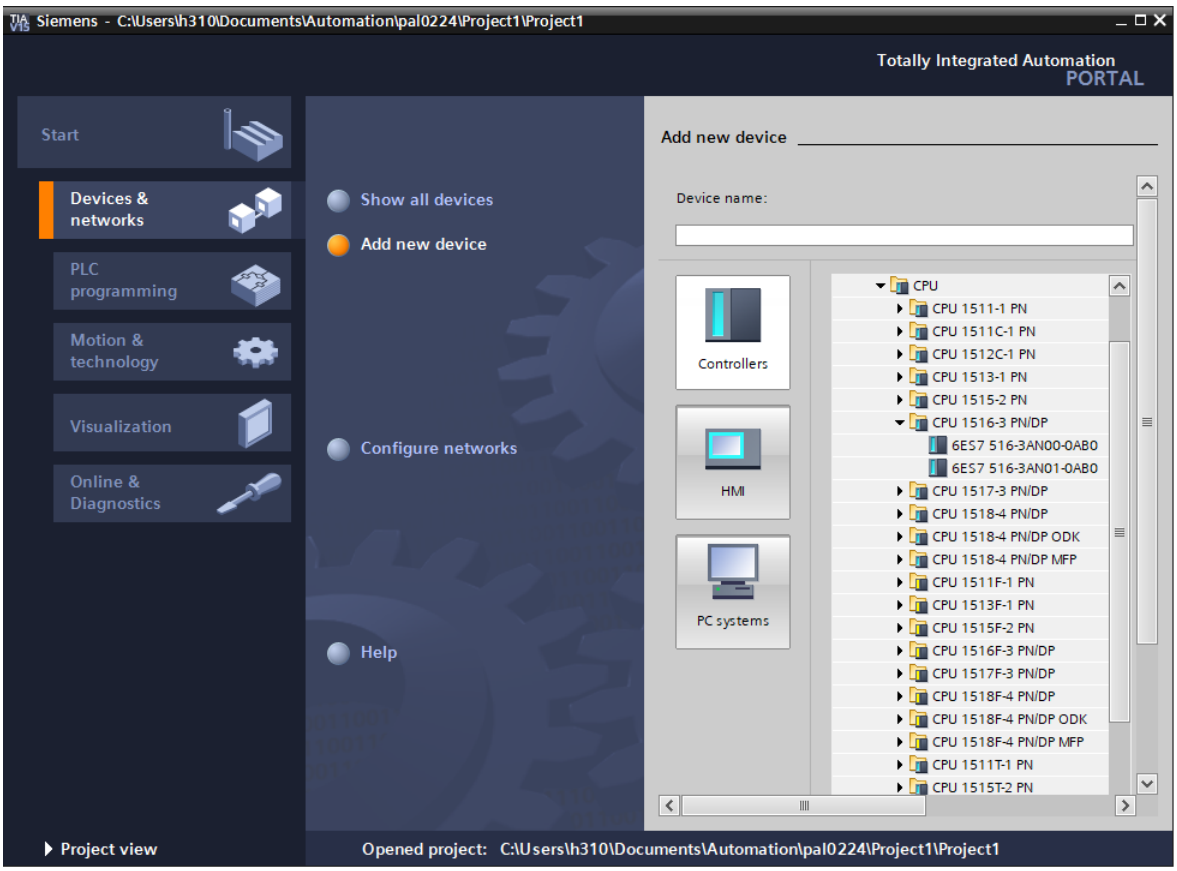


Fig. 10 CPU selection window

Running the hardware detection

1. Select the unspecified CPU in the project tree.
2. Select the "Hardware detection" function from the "Online" menu.
3. Select the "PN/IE" entry as the type of PG/PC interface.
4. Select the PG/PC interface.
5. Click the "Show all compatible devices" option.
6. Select the CPU from the compatible devices in the subnet.
7. Select the "Flash LED" checkbox to run a flashing test.
8. Click "Detect" to replace the unspecified CPU with the necessary CPU type.

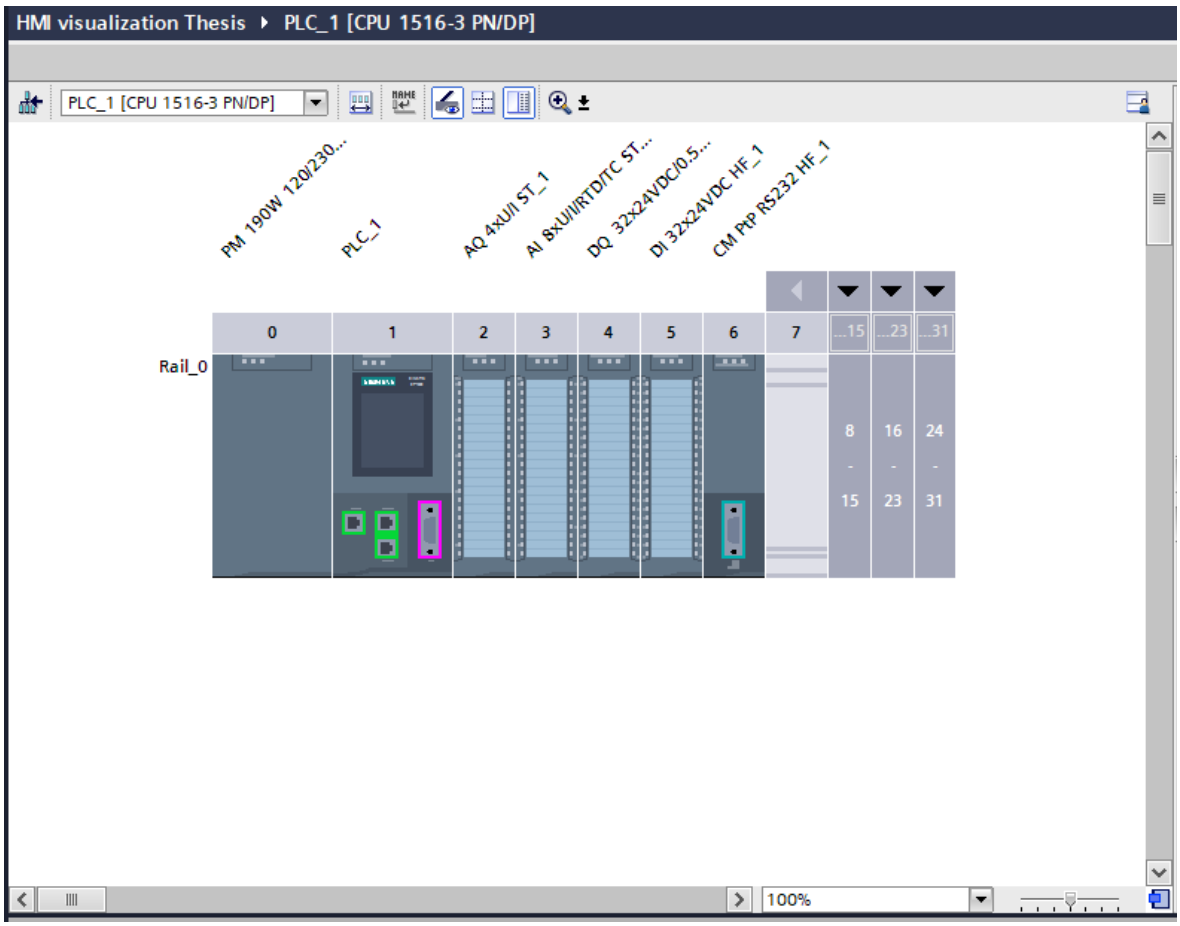


Fig. 11 CPU configuration window (Main window)

2.6 PLCSIM: (Simulator)

This is a simulation system that provides effective support with program development and specific applications. In an automation environment, ie., a simulated environment, the PLC and also the system start time is greatly reduced which in turn reduces the cost required to implement a real system as well as the cost of failure in case the system does not function properly. This makes it easier to detect program errors and help optimize program sections. If the program is changed, it is also possible to test it in this way before inserting it into the control system of the device. (Mrázek, 2016)

3 DESCRIPTION OF THE LABORATORY MODEL

In this chapter, the lab panel and model are described according to the specification which is present in the current laboratory setup.

3.1 Laboratory Panel

This model works as an interface between the PLC and the task being handled by the operator. It is divided into three parts, as can be seen in Fig. 12. On top of the module, we can see the digital inputs, digital outputs, analog inputs, and analog output ports. There are three functions for the toggle switches and it is used to set the digital input to either zero or one. If the toggle switch is in the lower position, that represents the logic "1". If it is in the middle position, then logic "0". When the switch is in the upper position, the digital inputs of the laboratory panel are getting supply from an external source. This panel also includes LEDs that display the state of digital inputs. (Mrázek, 2016)

Colour marking on the sockets

- Blue sockets are connected to the PLC output modules (both digital and analog).
- The red sockets represent supply voltage from a PLC or an external power source.
- Green sockets are connected to PLC input modules (digital and analog).
- The black sockets are grounded.

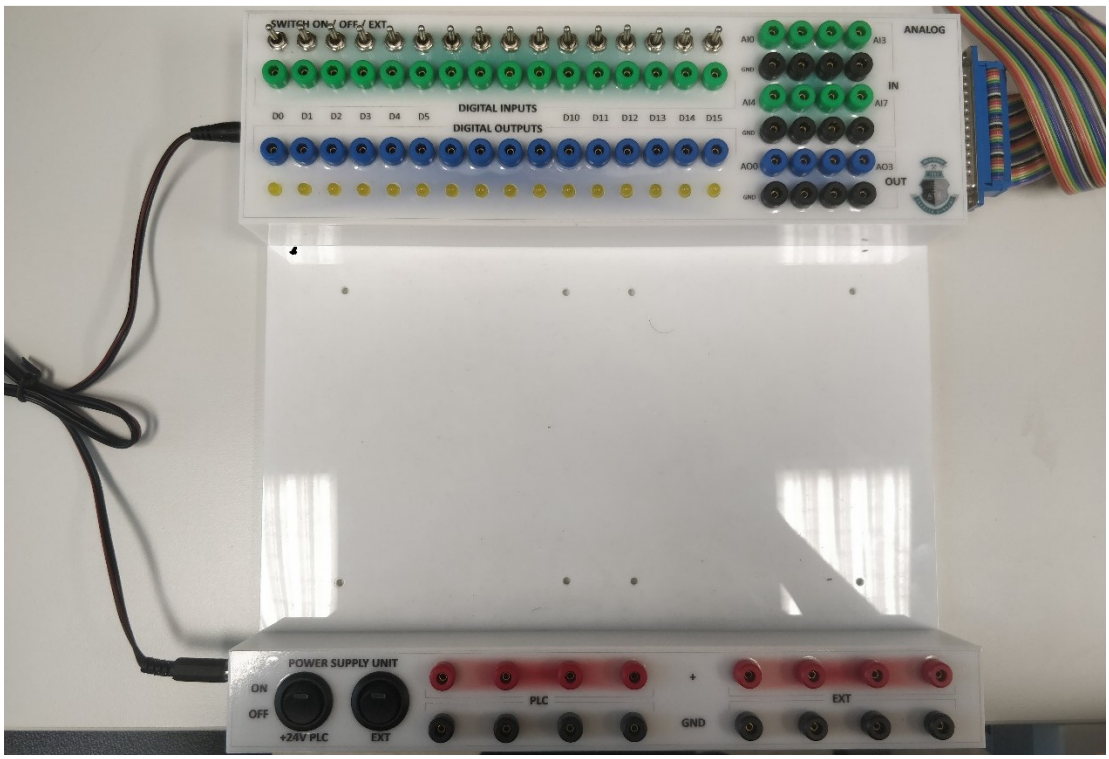


Fig. 12 Laboratory panel

On the left side, there is a power connector from the PLC to the bottom of the panel. A pair of connectors on the right are used to connect the sockets with the appropriate pins on the PLC. In the middle part of the panel, there is a free space to place the laboratory model and connect the model to the PLC. The left bottom section is serving as the power supply section, either as a 24 V DC supply from a PLC or from an external power supply. This part is equipped with on/off switches.

Tab. 1: Sockets in lab panel (Mrázek, 2016)

Sockets	I/O number in PLC	Sockets	I/O number in PLC
DI0	%I16.0	DQ6	%Q16.6
DI1	%I16.1	DQ7	%Q16.7
DI2	%I16.2	DQ8	%Q17.0
DI3	%I16.3	DQ9	%Q17.1
DI4	%I16.4	DQ10	%Q17.2
DI5	%I16.5	DQ11	%Q17.3
DI6	%I16.6	DQ12	%Q17.4
DI7	%I16.7	DQ13	%Q17.5
DI8	%I17.0	DQ14	%Q17.6
DI9	%I17.1	DQ15	%Q17.7
DI10	%I17.2	AI0	%IW6
DI11	%I17.3	AI1	%IW4
DI12	%I17.4	AI2	%IW2
DI13	%I17.5	AI3	%IW0
DI14	%I17.6	AI4	%IW14
DI15	%I17.7	AI5	%IW12
DQ0	%Q16.0	AI6	%IW10
DQ1	%Q16.1	AI7	%IW8
DQ2	%Q16.2	AQ0	%QW0
DQ3	%Q16.3	AQ1	%QW2
DQ4	%Q16.4	AQ2	%QW4
DQ5	%Q16.5	AQ3	%QW8

3.2 Laboratory Model

The model is consisting of a 125 x 150 mm panel on which the sockets leading to the individual elements are placed. These sockets are used to connect with the laboratory panel.

Major parts:

1. Bulb
2. Temperature sensor
3. Fan(ventilator)

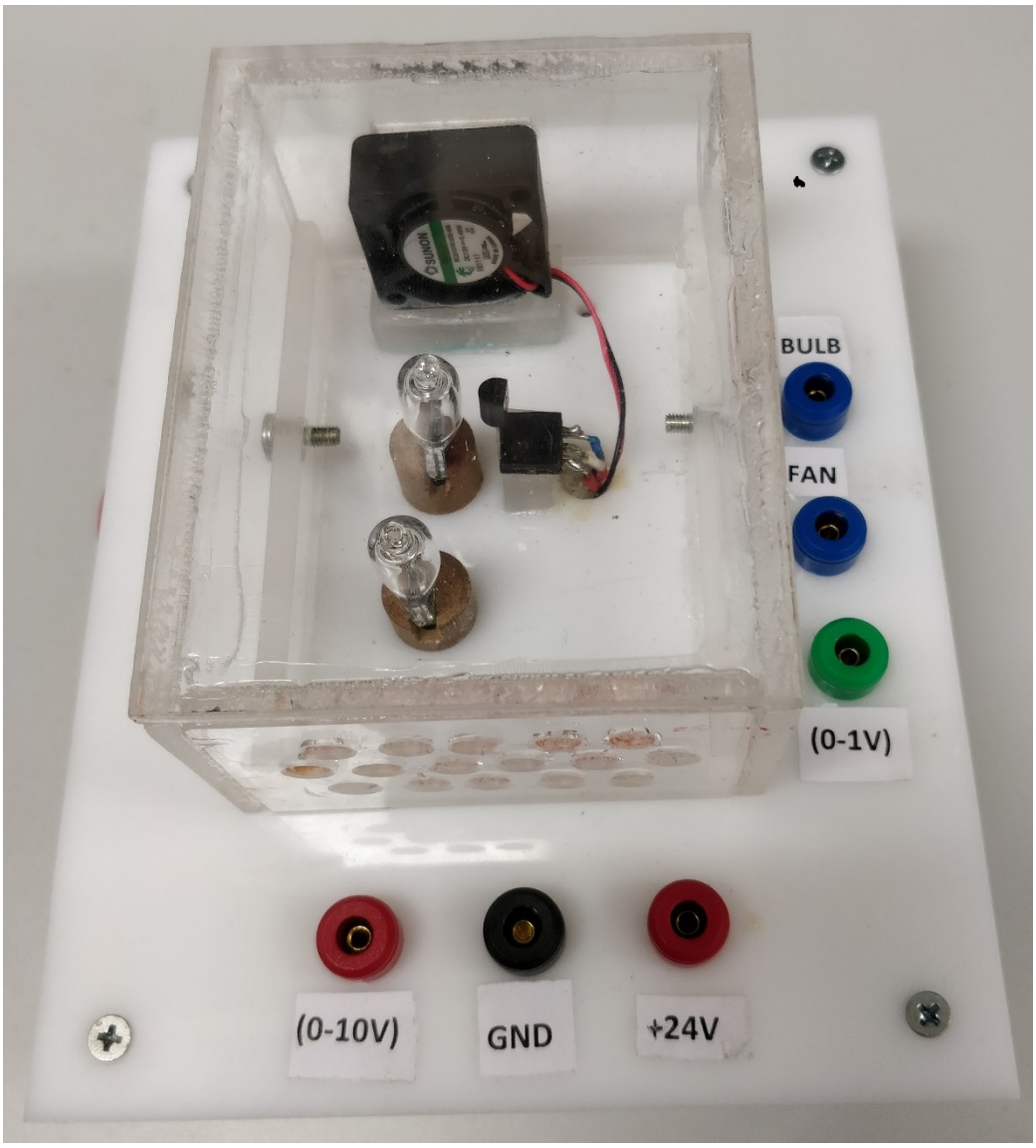


Fig. 13 Laboratory model

3.2.1 Bulb

I have used 2 halogen bulbs in the laboratory model. Each bulb is 12V-powered with a power of 10W and it will serve as the temperature source. Two bulbs are connected serial connection and it will act as a control element. It will be controlled by the analog output of the PLC (0-10V).

3.2.2 Temperature sensor

The TMP36 (low-temperature temperature) sensor will be used to measure the source temperature

- Supply voltage ranging from 2.7V to 5.5V
- Calibrated in ° C with a scale of 10 mV / ° C.
- Sensor range -40 ° C to 150 ° C.

The temperature sensors will be located away from the heat source in order to measure the temperature of the heat source as accurately as possible. But it should be placed in sufficient distance to avoid damage to the sensor since the source temperature is very high. The TMP36 is specified from −40°C to +125°C, provides a 750mV output at 25°C. To obtain the temperature from the measured value, the formula must be used:

$$\text{Temp in °C} \approx \frac{[(\text{Vout in mV}) - 500]}{10} \tag{3.1}$$

3.2.3 Fan (Ventilator)

As another element on the model, the Sunon model MC25100V2-A9 fan is used.

- Rated speed 10000 rpm.
- equipped with a ball bearing
- Operated voltage range 2.5-6 V DC.
- Operating temperature range -10 ºC to+70 ºC.
- 12V DC and 72 mA power supply.
- Dimensions 25 × 25 × 10 mm.
- Airflow is 3 CFM (cubic feet per minute).

The fan purpose is to cool the source and the temperature sensor, act as a disturbance in the temperature measurement system, by the air flow between the heat source and the temperature sensor. The fan will be controlled via an analog output of the PLC (0-10V).

4 ELECTRICAL DESIGN

In below, we can see the electrical circuit is designed for fan, bulb and temperature sensor. Operational amplifier and transistor are attached in the fan and bulb circuit, which helps to convert 24V PLC power supply into 0-12V. Both fan and bulb are controlled by 0-10V from the PLC. Both the amplifier and transistor are triggered by the external voltage supply of 24V, which is given by the external power resource.

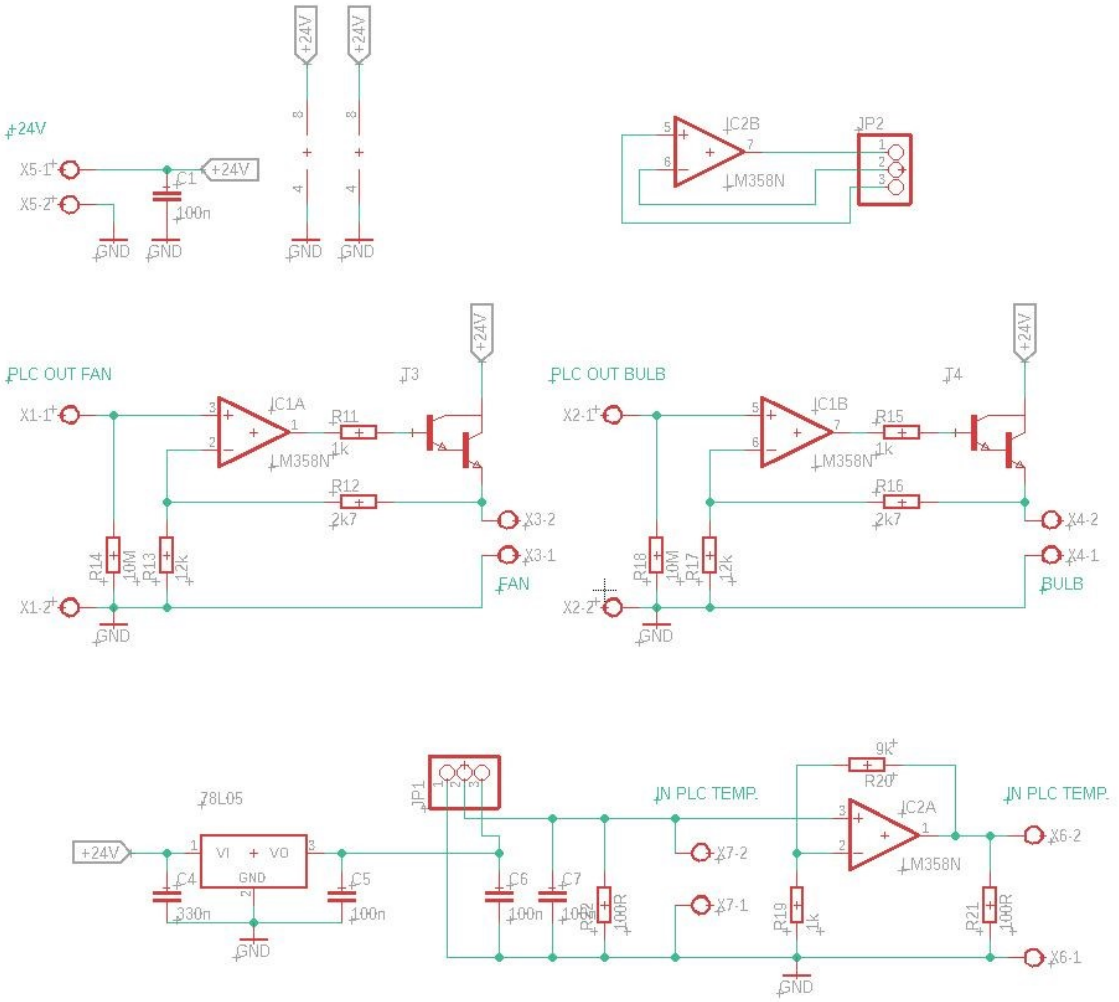


Fig. 14 Circuit of laboratory model

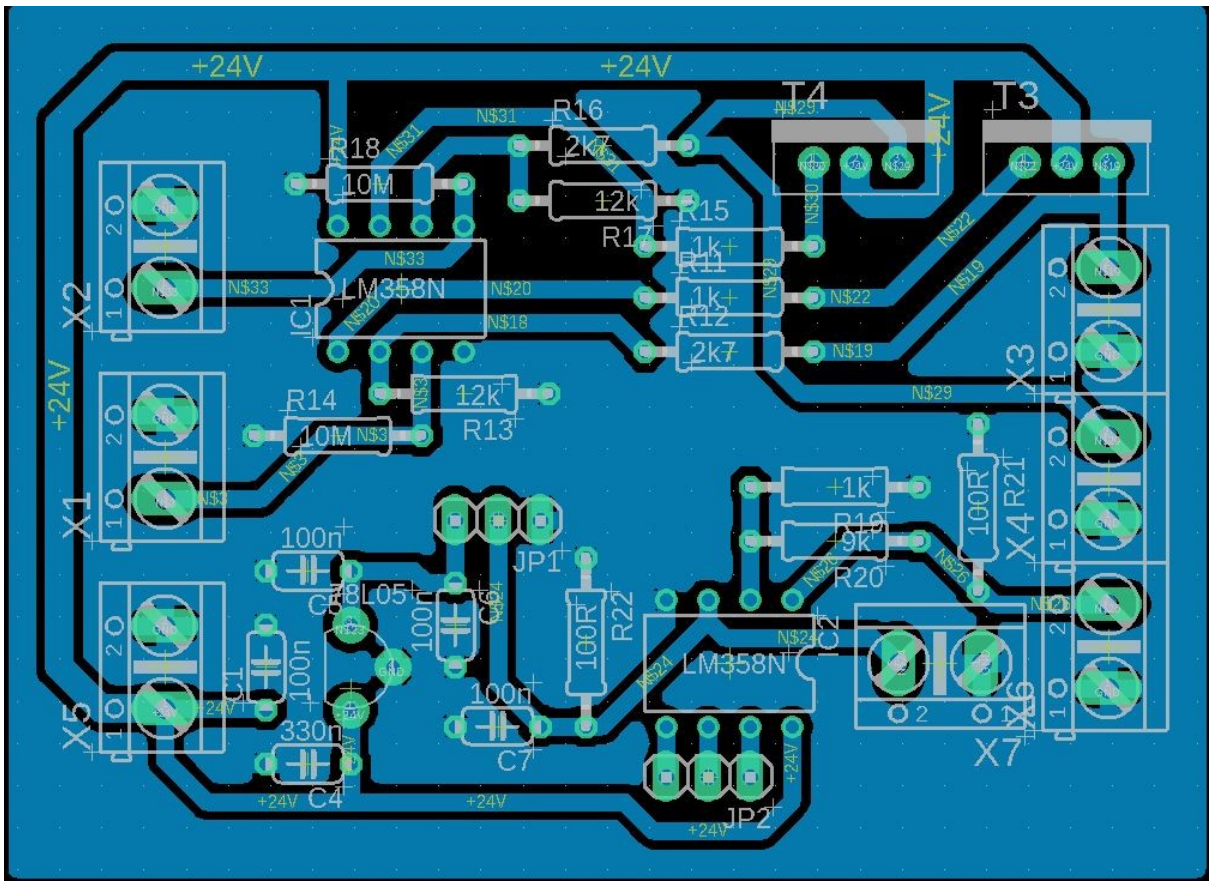


Fig. 15 PCB board design

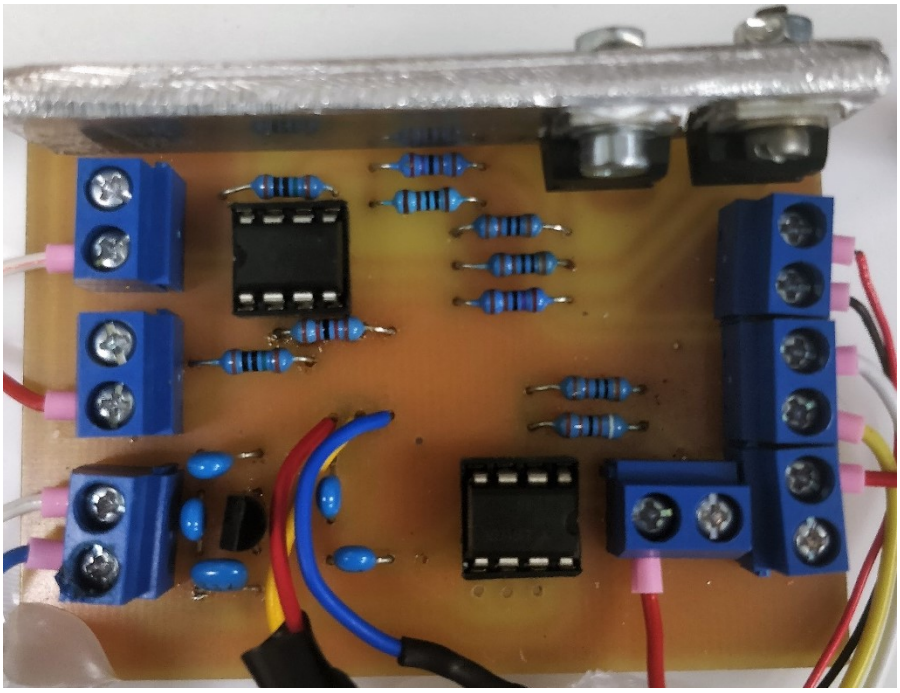


Fig. 16 PCB board

Output pins of both fan and bulb will produce 0-12V supply. An external voltage of 24V is also given to the voltage regulator. The main role of the regulator is to convert external 24V into 5V and the converted voltage is given to temperature sensor TMP36. And then, the

output from the sensor is amplified and temperature measured in voltage, later it is converted into degree Celsius. The capacitor is connected in the input and output of both voltage regulator and temperature sensor to smoothen the voltage. Specification of the circuit is mentioned in the previous chapter.

5 SYSTEM IDENTIFICATION

Identifying and predicting the nature of the system are documented in this following chapter. In general, the system can be identified in an open loop system. In the open loop system, the input is given to the plant (laboratory model) and output of the step response measured graphically by using traces. Using these traces, we can identify whether the system is proportional, integral or derivative. And these traces are used to identify the transfer function of the system.

In my case, I have an input to the bulb (0-10V) and measured the step response of the plant by using the temperature sensor (TMP36).

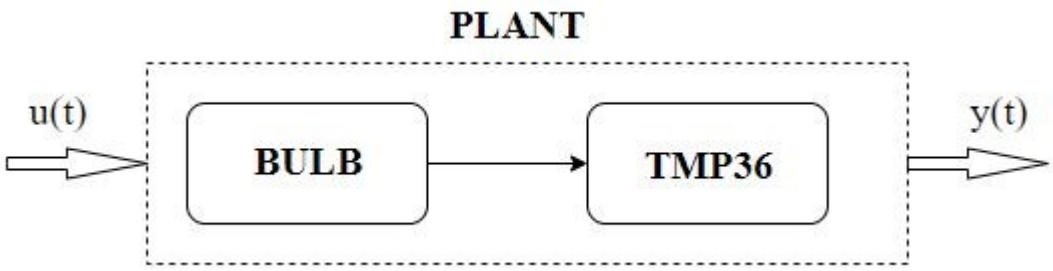


Fig. 17 Open-loop control system

Step response of the system is implying the system is proportional with the first order of inertia. The step response of the system is shown in the picture below.

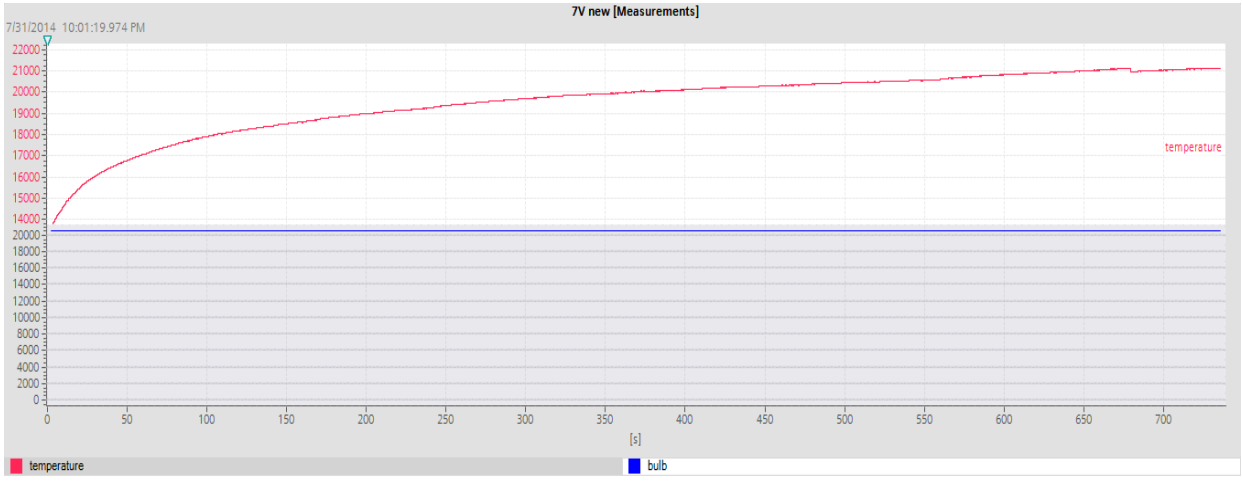


Fig. 18 Step response of the system in open-loop

The transfer function of the system can be found out by using standard methods like the one-point method and the two-point method, etc.

The stated transfer function is identified by using the one-point method.

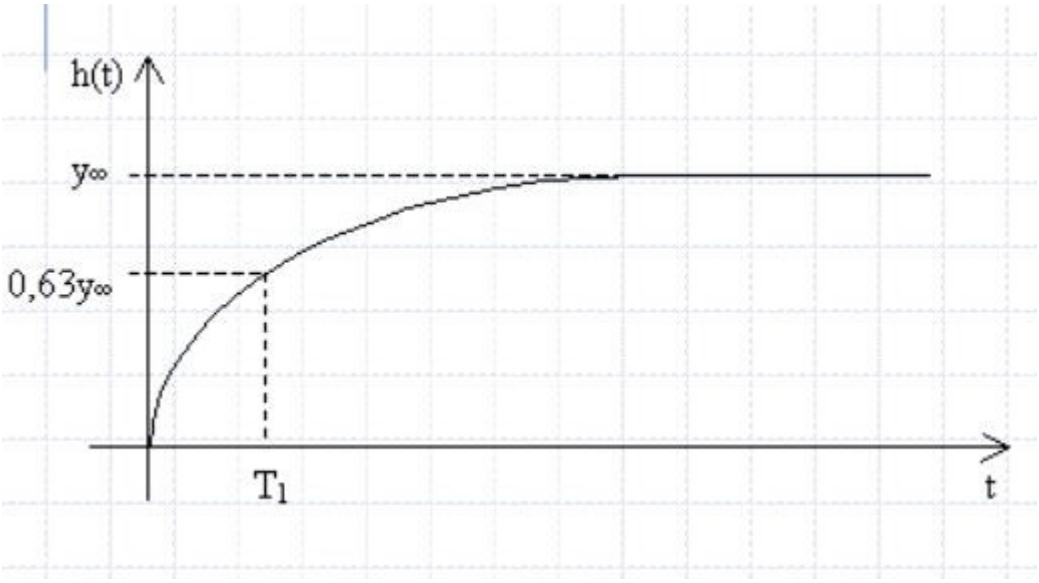


Fig. 19 One-point approximation method (Wagnerova, 2019)

Transfer function:

$$G(s) = \frac{k}{T_1 S + 1} \tag{5.1}$$

$$0.63y(\infty) = t_{0.63}$$

$$T_1 = t_{0.63} \tag{5.2}$$

$$k = \frac{y(\infty)}{u(\infty)} \tag{5.3}$$

The transfer function of the system is

$$G(s) = \frac{0.2757}{114.854S + 1}$$

$$k = \frac{1.930}{7} = 0.2757$$

By using this transfer function, we can find the control parameters K_P, T_I, and T_D. Tuning parameter or control parameter can be found by both theoretical and practical method. The desired model method or universal experimental method is used for theoretical process and MATLAB controller tuning is a practical method.

The transfer function of the system is evaluated in the MATLAB Simulink. In MATLAB Simulink, the transfer function is created and executed in both open loop and closed loop method.

By seeing the graph of the open loop, we would be able to identify the value of y (∞) =1.930.

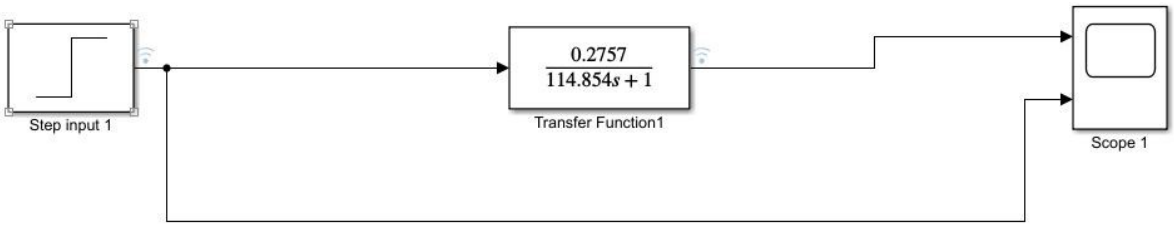


Fig. 20 Open-loop process for system identification

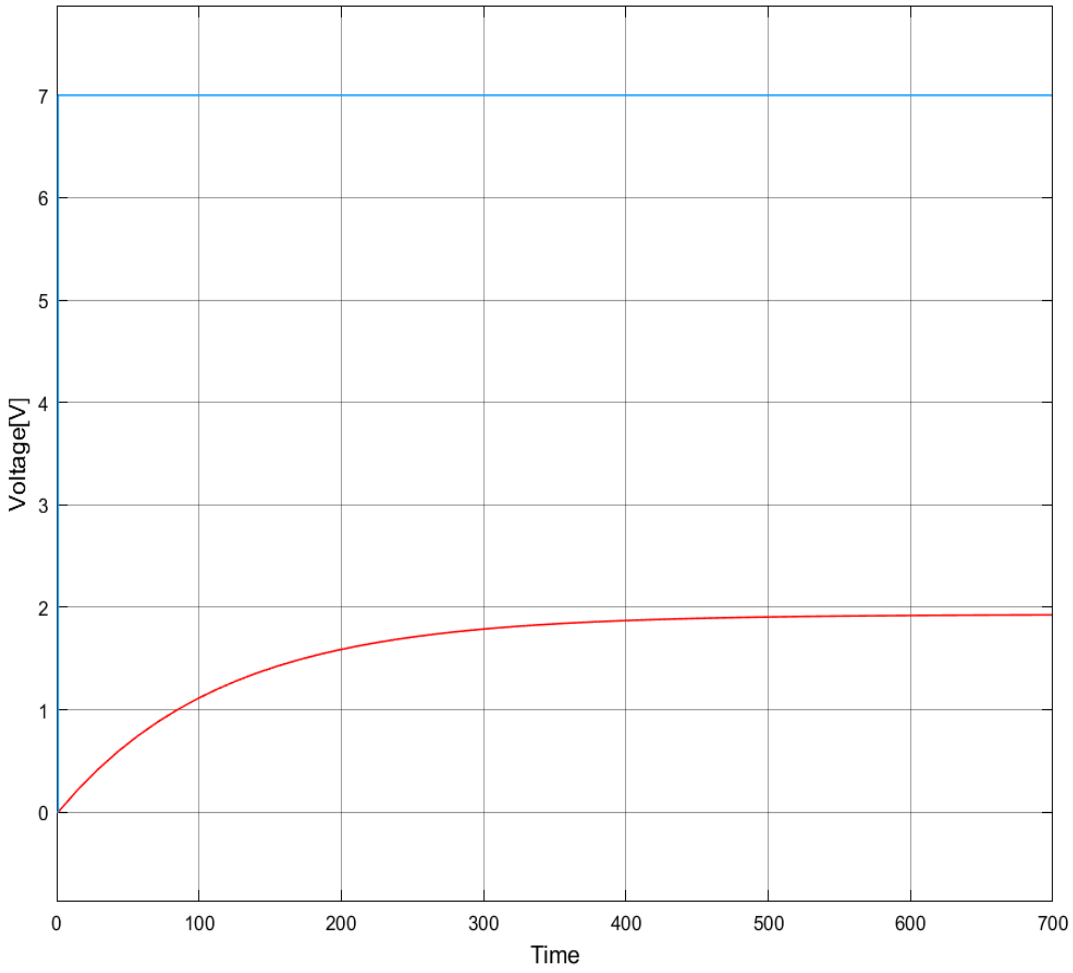


Fig. 21 The response of the open-loop process used for system identification

In the closed-loop system, the controller is introduced. By using the tuner present in the Simulink, we can adjust the system response as without overshoot and obtain the tuning parameters (K_P, T_I and T_D).

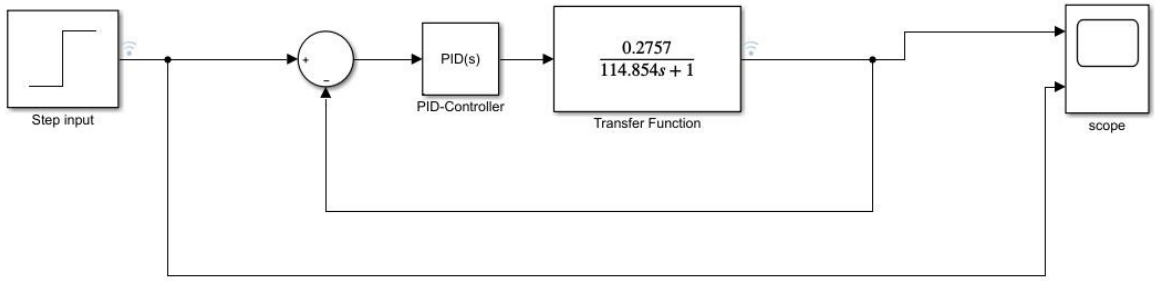


Fig. 22 Closed-loop process

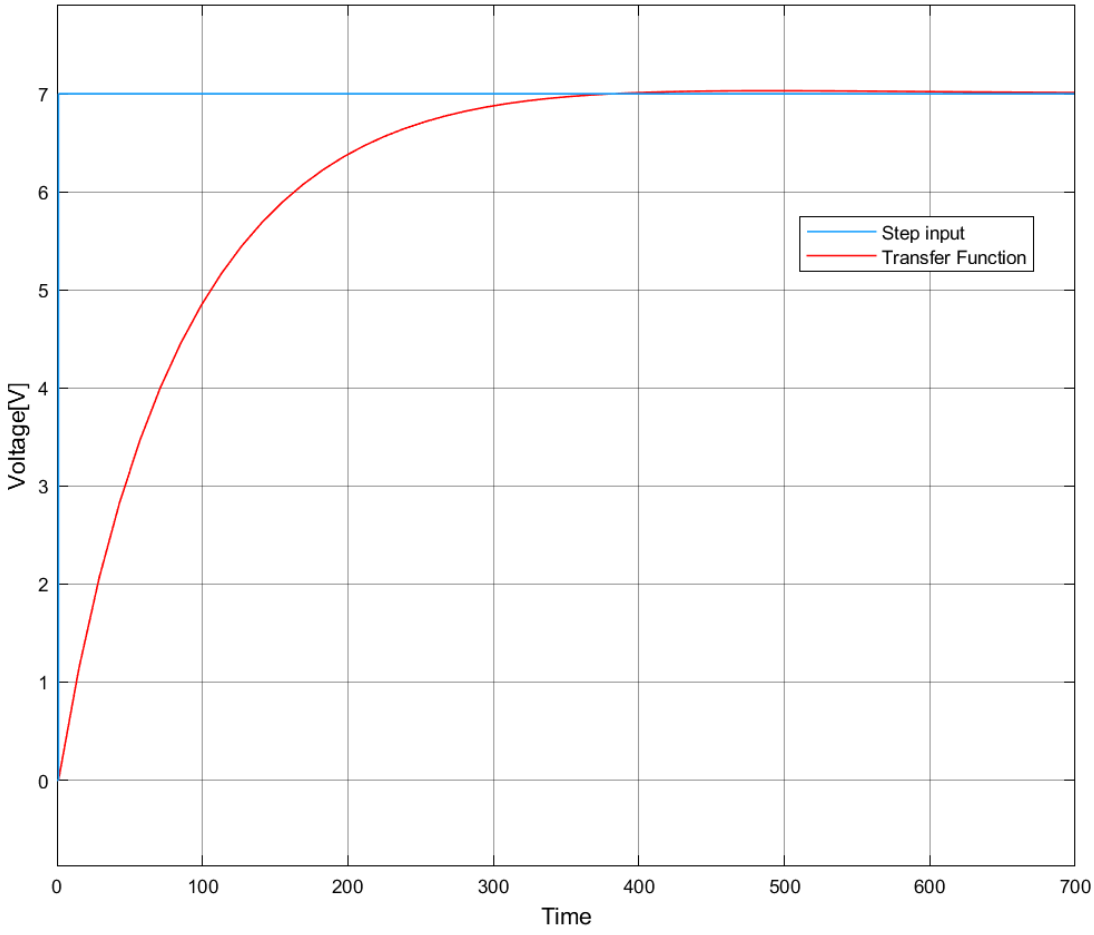


Fig. 23 The response of closed-loop process

The value of the control parameters is

PI controller: K_P = 10.23837[s], T_I = 1.148540[s]

PID controller: K_P = 5.160681[s], T_I = 1.0021154[s], T_D =0.07347878[s]

Results for these parameters are explained in the upcoming chapters. For these types of response, the PI controller is the most suitable controller.

6 CREATION OF CONTROL ALGORITHM IN TIA PORTAL V15

After configuring the S7-1500 device in the TIA Portal. We should program the algorithm for both open loop and closed loop control. In the programming blocks, we should create some program blocks and data blocks for preparing this control algorithm.

Programming and data blocks:

- Start-up
- Main-loop
- Cyclic interrupt
- PID (Functional block)
- Variables (Data block) and PLC tags

6.1 Start-up

You can specify the conditions for starting up your CPU (initialization values for RUN, start-up values for I/O modules) by writing your program for the start-up in the organization blocks OB100 for restart (warm restart), OB101 for hot restart, or OB102 for cold restart. There are no restrictions to the length of the start-up program and no time limit since the cycle monitoring is not active. During the start-up, all digital outputs have the signal state 0.

In the start-up cycle, I initialized QW0 (bulb analog value) and QW2 (an analog value) to be 0 and K_P, T_I, and T_D to be '1'. This initialization will feed to the respective tags, every time when the system restarts.

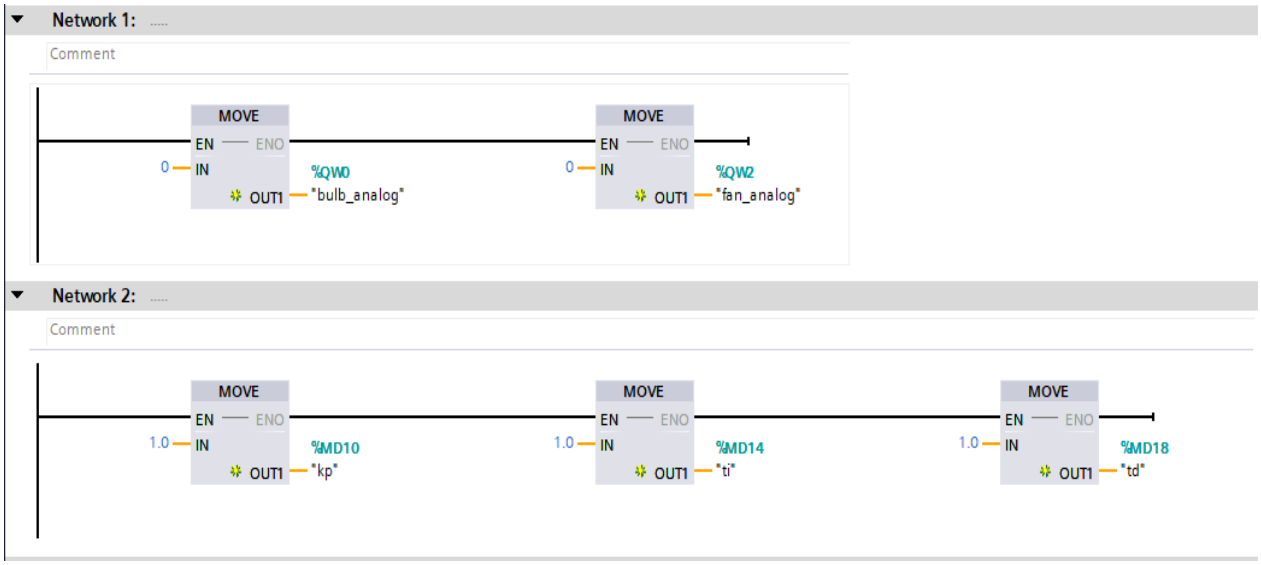


Fig. 24 Start-up cycle

6.2 Main-loop

Organization blocks (OBs) represent the interface between the operating system and the user program. Called by the operating system, they control cyclic and interrupt of program execution, start-up behaviour of the PLC and error handling. You can program the organization blocks to determine CPU behaviour.

In the organization block, programming for open loop voltage scaling of bulb and fan closed loop set point and recalculation of input voltage to the analog value.

By giving voltage value to "variable bulb open loop voltage "and "variable fan open loop voltage ", it will automatically be recalculated to an integer value with the help of calculation block and the value will be fed into MW0 and MW2 respectively.

In network 2 and 3, we can see the program for restriction of input voltage for the user. It will help and maintain maximum value at 10. Setpoint MD22 is used for closed-loop control. Setpoint is given in terms of ºC. According to the value of setpoint, the system will try to adjust and maintain the temperature.

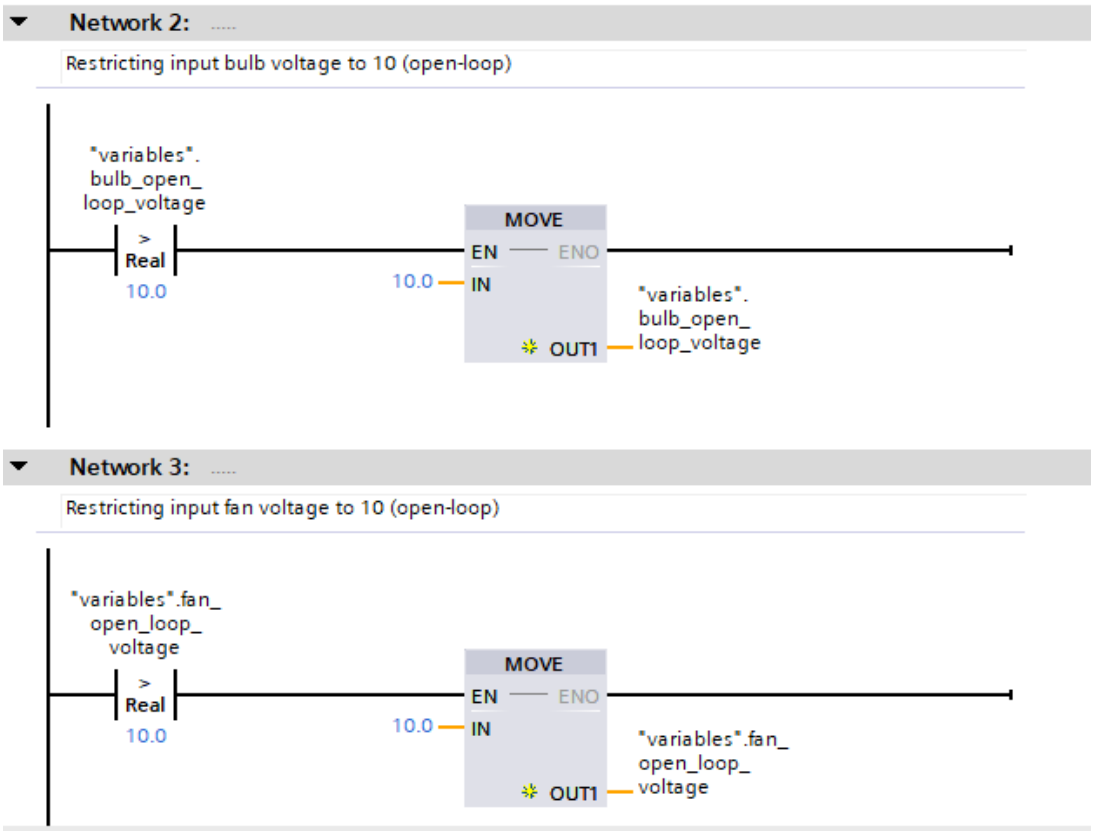


Fig. 25 Voltage restriction of open-loop (Main-loop)

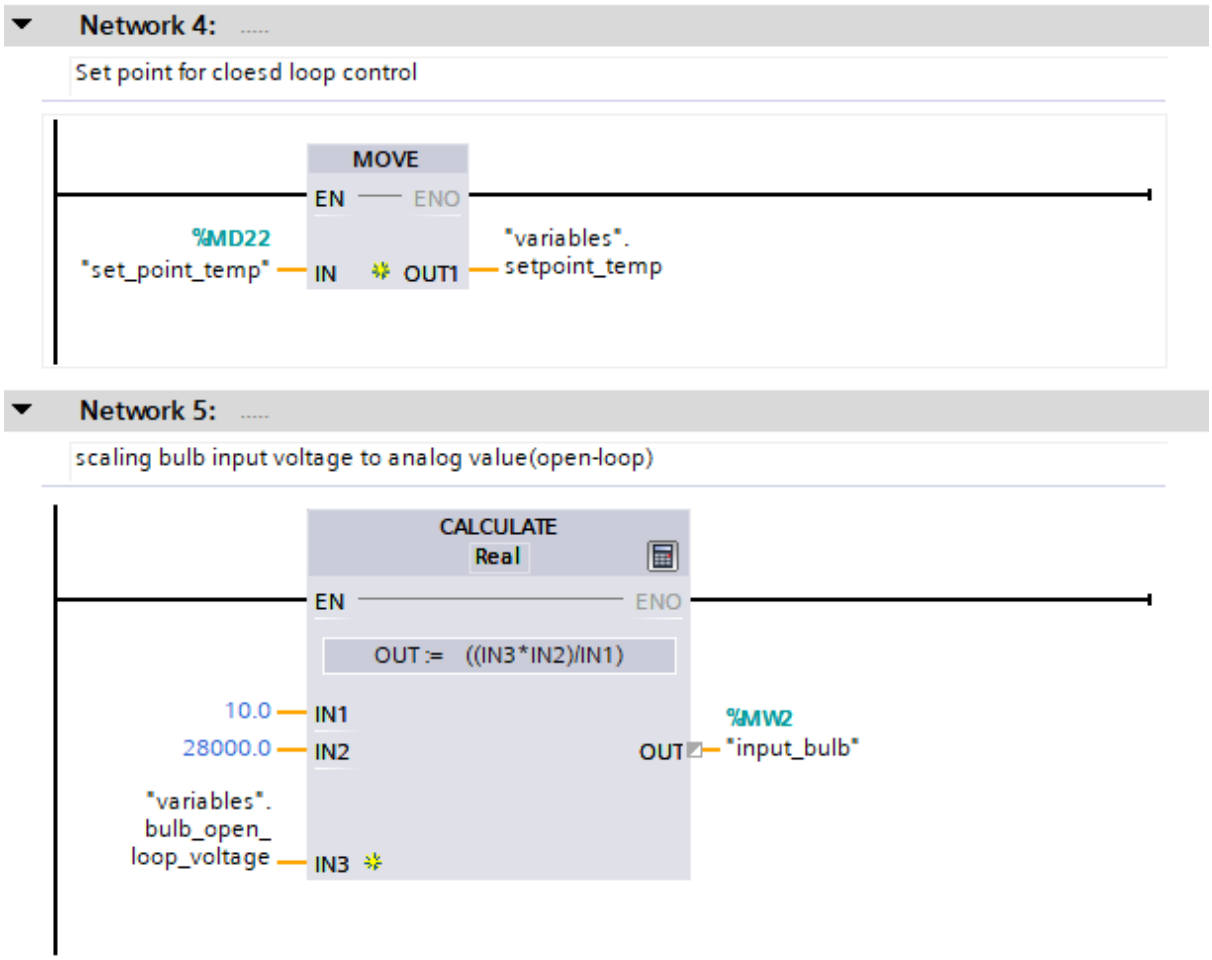


Fig. 26 Setpoint temperature and recalculation of voltage to analog value for the bulb (Main-loop)

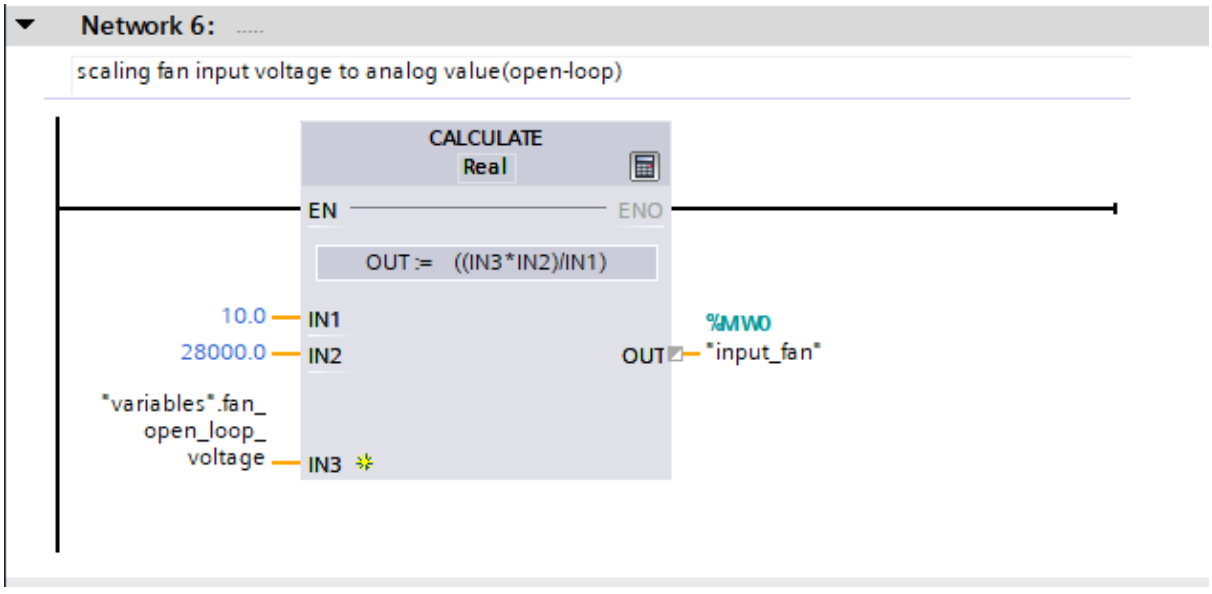


Fig. 27 Recalculation of voltage to analog value for the fan (Main-loop)

Recalculation of voltage into analog value to feed input for both fan and bulb,

$$\text{Analog value} = \frac{\text{input voltage (V)} * 28000}{10}$$

The value 28000 is maximum analog value of the bulb for 10V.

6.3 Cyclic interrupt

The S7 CPUs provide cyclic interrupt OBs that interrupt cyclic program processing at certain intervals.

Cyclic interrupts are triggered at intervals. The time at which the interval starts is the mode transition from STOP to RUN.

In the cyclic interrupt block, Control algorithm of both open loop and closed loop system is created. We can measure the step response of the system by using the temperature sensor (connected in IW0).

Since the open loop recalculation takes place in the main loop, it will automatically feed the values to MW0 and MW2. Once M16.0 and M16.1 are turned on with the help of the visualization button, both bulb and fan will turn on. By using temperature sensor analog value, we can measure the step response characteristic.

Initially, the integer value of the temperature sensor is recalculated to degree Celsius. To calculate the error, I converted both setpoint temperature and real-time temperature into a voltage.

The error value is stored in "variables error voltage" and moved to "variables previous error voltage" for calculating T_D in the PID controller.

The parameter of PID is fed into the controller and the controller starts to calculate the output voltage. Once the output voltage is calculated, it will be automatically recalculated to the integer value and fed into the analog output of the bulb.

This process will work as a closed loop system to maintain the box temperature according to the setpoint.

PID controller can be turned on and off by using M16.2 bit. If the PID controller in the off state, "variables bulb output voltage" will be set to zero. This condition will stop the closed-loop system if PID is in the off state.

The interlocking system is done in this cyclic interrupt, we cannot turn on both open loop and closed loop at the same time. It may cause extreme damage to the system.

So closed contact of bulb switch (M16.0) is given to the closed loop and respectively the closed contact of PID switch (M16.2) is given to open loop.

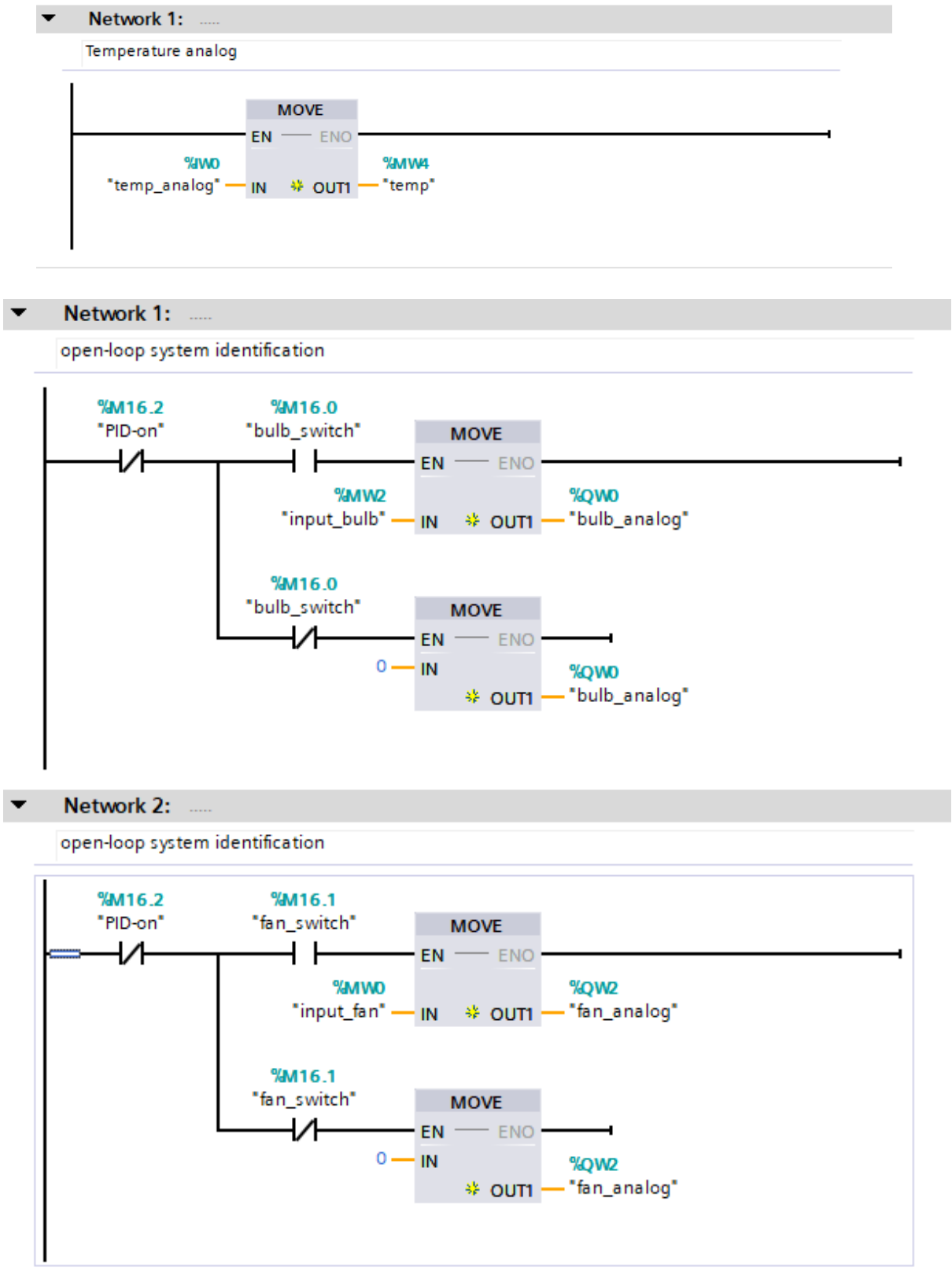


Fig. 28 Temperature analog value control algorithm for open-loop (cyclic interrupt)

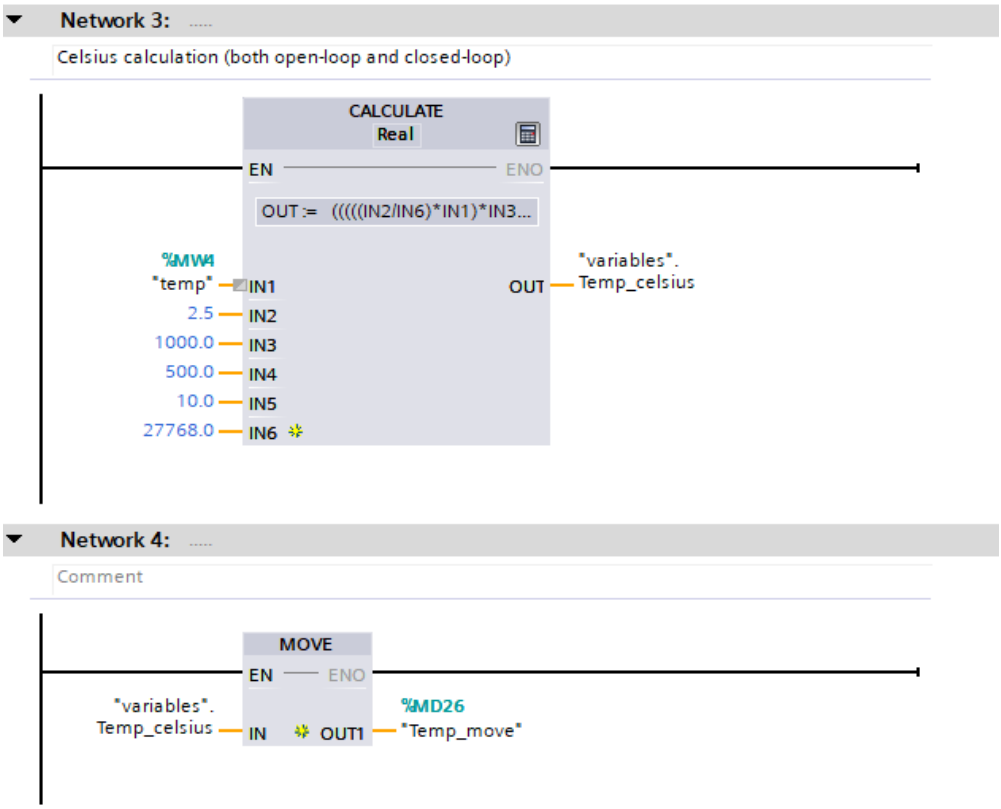


Fig. 29 Recalculation integer value to degree Celsius and temp move for data logging (cyclic interrupt)

To calculate the temperature of the system,

$$\text{voltage} = \frac{2.5}{27768} \times \text{Measuring ADC value[V]}$$

$$\text{Temp in °C} \approx \frac{[(\text{Vout in mV}) - 500]}{10}$$

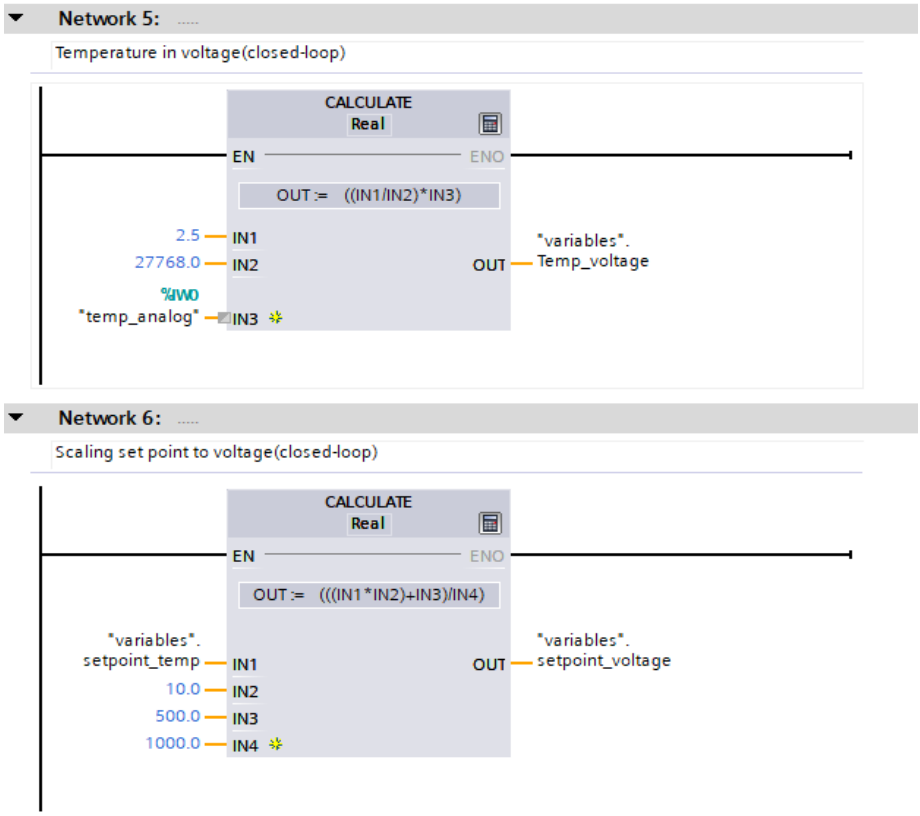


Fig. 30 Setpoint voltage and temperature voltage calculation (cyclic interrupt)

Converting temperature analog value into a voltage to feed the error,

$$\text{Temperature [V]} = \frac{2.5}{\text{analog value of temperature}} * 27768$$

Setpoint temperature value converted into voltage,

$$\text{Setpoint [V]} = \frac{\big((\text{setpoint in celsius}) * 10\big) + 500}{1000}$$

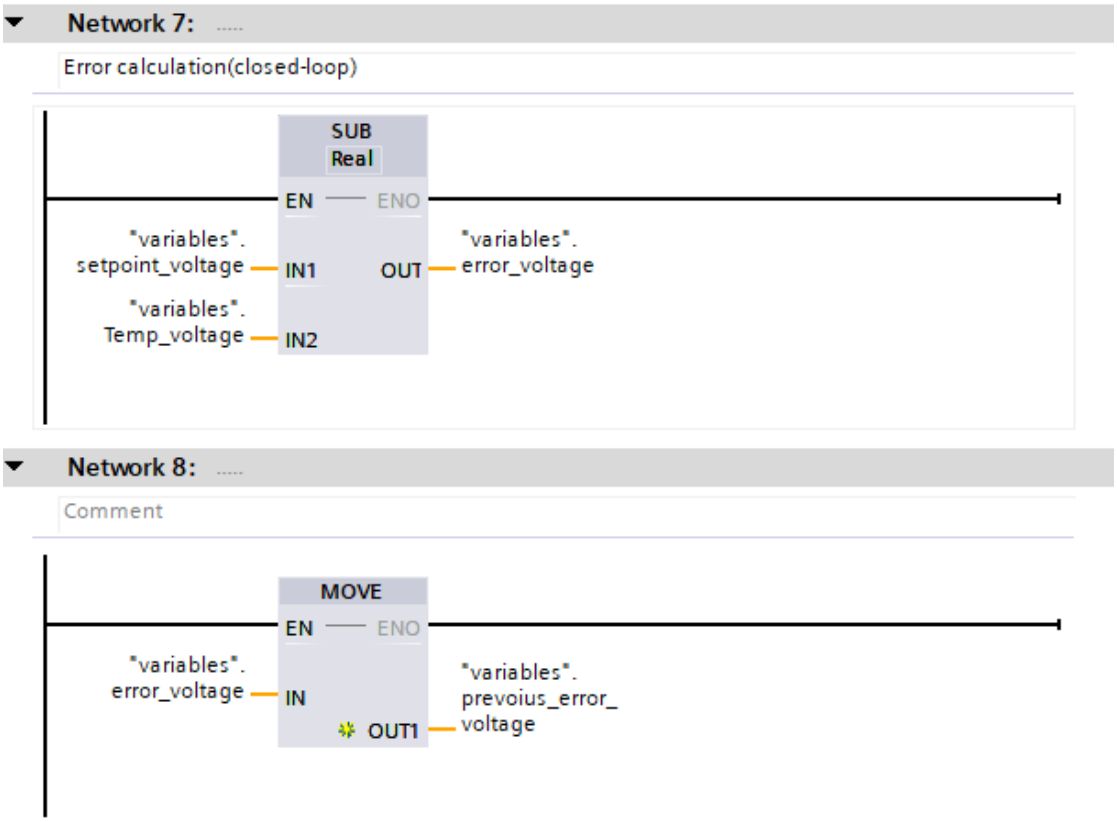


Fig. 31 Error voltage calculation (cyclic interrupt)

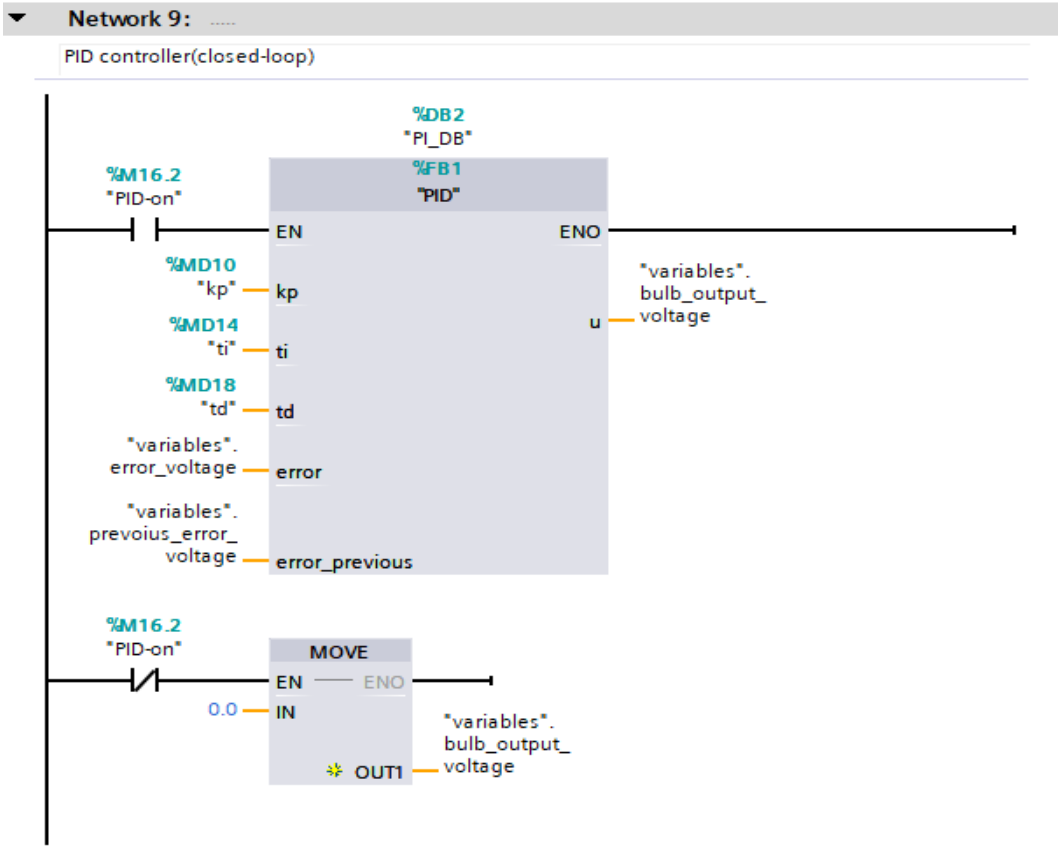


Fig. 32 PID controller (cyclic interrupt)

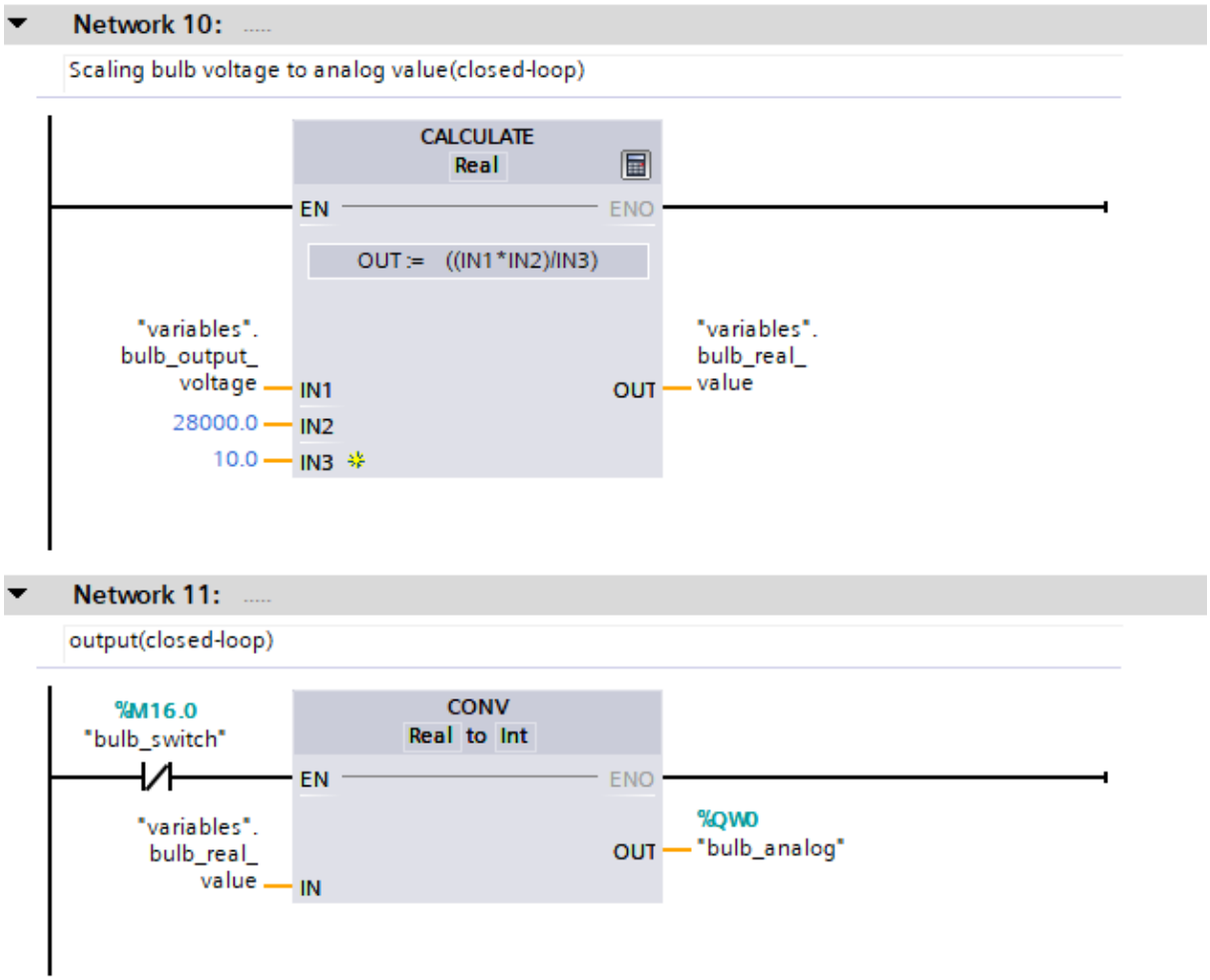


Fig. 33 Controller voltage recalculation to integer (cyclic interrupt)

Recalculation of controller voltage into anlaog value to feed the bulb,

$$\text{bulb analog} = \frac{\text{controller voltage} * 28000}{10}$$

6.4 PID (Functional block)

Function blocks are code blocks that store their output, input and in-out parameters permanently in data blocks so that they remain available even after the block has been executed. Therefore, they are also referred to as blocks "with memory".

Function blocks can also operate with temporary tags. Temporary tags are will not be stored in the instance DB, but are available for one cycle only.

FC Blocks are part of the program called from within the main program. For example, instead of typing multiple times in the main loop, an operation that is to be repeated more than once is written into the FC block, and only this block is called every time it is needed. A written program is shorter.

My functional block is based on SCL Language. In order to create my own PID controller, SCL language is used in my functional block.

```
IF...  CASE... OF...  FOR... TO DO..  WHILE.. DO...  (*...*)  REGION

 1 IF (#u >= 10 AND #error > 0) OR (#u <= 0 AND #error < 0) THEN
 2     #error_sum := #error_sum;
 3 ELSE
 4     #error_sum := #error_sum + #error;
 5 END_IF;
 6
 7 IF #ti = 0 AND #td = 0 AND #kp = 0 THEN
 8     #u := 0;
 9 ELSIF #ti = 0 AND #td = 0 THEN
10     #u := #kp * #error;
11 ELSIF #ti = 0 THEN
12     #u := #kp * (#error + ((#td / #ts) * (#error - #error_previous)));
13 ELSIF #td = 0 THEN
14     #u := #kp * (#error + (#ts / #ti) * #error_sum);
15 ELSE
16     #u := #kp * (#error + ((#ts / #ti) * #error_sum) + ((#td / #ts) * (#error - #error_previous)));
17
18 END_IF;
19
20 IF #u < 0 THEN
21     #u := 0;
22 ELSIF #u > 10 THEN
23     #u := 10;
24 END_IF;
25
26
```

Fig. 34 SCL programming of PID controller (Functional block)

In the above code, I designed the controller as operator friendly. The operator can choose the type of the controller, it may be P, PI, PD, or PID.

The first 5 lines of the code are for Anti-windup effect and the last 5 lines of code are to restrict the value of the voltage between 0-10V. All input and output values of the PID controller is created in Real data type. Inputs are K_P, T_I, T_D, error and previous error and outputs are a voltage (u).

T_s is sample time and it is set to the default value of 0.01 and error sum is created in static to store and reuse the value every time in the code.

Digital PID controller is

$$u(kT) = K_p\left\{e(kT) + \frac{T}{T_i}\sum_{i=o}^{k} e(iT) + \frac{T_D}{T}\{e(kT) - e[(k-1)T]\}\right\} \quad (4.1)$$

K = 0,1,2,3…. kT is discrete time, T is sampling period.

	Name	Data type	Default value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Input									
2	kp	Real	0.0	Non-retain	☑	☑	☑			
3	ti	Real	0.0	Non-retain	☑	☑	☑			
4	td	Real	0.0	Non-retain	☑	☑	☑			
5	error	Real	0.0	Non-retain	☑	☑	☑			
6	error_previous	Real	0.0	Non-retain	☑	☑	☑			
7	<Add new>									
8	Output									
9	u	Real	0.0	Non-retain	☑	☑	☑			
10	InOut									
11	<Add new>									
12	Static									
13	error_sum	Real	0.0	Non-retain	☑	☑	☑			
14	Temp									
15	<Add new>									
16	Constant									
17	ts	Real	0.01							

Fig. 35 Data's used for designing a PID controller (Functional block)

6.5 Variables (data block) and PLC tags

The data block is a well-structured memory area and we can access the data at high speed. All memory parts of data can be called globally anywhere in the program. The structure of the global DB's is made up of all data types.

I named my data blocks as “variables”. Important parameters of the system are stored and called from DB.

PLC tags is also a memory for switches, input and output purposes. QW0, QW2, and IW0 are part of PLC tags.

Pictures of Data blocks and PLC tags are mentioned below.

variables

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Static									
2	setpoint_temp	Real	0.0		☑	☑	☑			
3	Temp_celsius	Real	0.0		☑	☑	☑			
4	Temp_voltage	Real	0.0		☑	☑	☑			
5	setpoint_voltage	Real	0.0		☑	☑	☑			
6	error_voltage	Real	0.0		☑	☑	☑			
7	prevoius_error_voltage	Real	0.0		☑	☑	☑			
8	bulb_output_voltage	Real	0.0		☑	☑	☑			
9	bulb_real_value	Real	0.0		☑	☑	☑			
10	bulb_open_loop_volt...	Real	0.0		☑	☑	☑			
11	fan_open_loop_voltage	Real	0.0		☑	☑	☑			

Fig. 36 “Variables" (data block)

PLC tags

	Name	Tag table	Data type	Address	Retain	Acces...	Writa...	Visibl...	Supervis...	Comment
1	bulb_analog	Default tag table	Int	%QW0	☐	☑	☑	☑		
2	fan_analog	Default tag table	Int	%QW2	☐	☑	☑	☑		
3	temp_analog	Default tag table	Int	%IW0	☐	☑	☑	☑		
4	input_bulb	Default tag table	Int	%MW2	☐	☑	☑	☑		
5	input_fan	Default tag table	Int	%MW0	☐	☑	☑	☑		
6	temp	Default tag table	Int	%MW4	☐	☑	☑	☑		
7	set_point_temp	Default tag table	Real	%MD22	☐	☑	☑	☑		
8	bulb_switch	Default tag table	Bool	%M16.0	☐	☑	☑	☑		
9	fan_switch	Default tag table	Bool	%M16.1	☐	☑	☑	☑		
10	kp	Default tag table	Real	%MD10	☐	☑	☑	☑		
11	ti	Default tag table	Real	%MD14	☐	☑	☑	☑		
12	PID-on	Default tag table	Bool	%M16.2	☐	☑	☑	☑		
13	td	Default tag table	Real	%MD18	☐	☑	☑	☑		
14	Temp_move	Default tag table	Real	%MD26	☐	☑	☑	☑		
15	bulb_analog_1	Default tag table	Int	%QW4	☐	☑	☑	☑		
16	open_loop_log	Default tag table	Bool	%M16.3	☐	☑	☑	☑		
17	closed_loop_log	Default tag table	Bool	%M16.4	☐	☑	☑	☑		

Fig. 37 PLC tags

7 HMI

HMI is a human-machine interface. It is an integral part of the SCADA system. It allows to communicate between human operators and machines. HMI provide the means by which process personnel interacts with the PLC control system. A well-designed combination of HMI's and PLC's can be a solid foundation for process automation needs.

7.1 TP700 Comfort

TP700 HMI comfort provides operators or workers a way to communicate and manage with a system. This communication is through a special graphical user interface (GUI), which allow us to exchange information and communication between PLC and machine. HMI software is completely designed for both machine level and supervisory level.

SIMATIC HMI products are the intelligent response to complex processes and stringent requirements for the operation of machines and plants.

Fig. 38 TP700 comfort panel *(SIEMENS[10], 2018)*

7.1.1 Advantages (SIEMENS[9], 2018)

1. Available in different sizes
    - Screen sizes in 4‘‘, 7‘‘, 9‘‘, 12‘‘and 15‘‘with keys or touch and 19‘‘and 22‘‘touch; the 4‘‘available with additional keys.
    - The devices can be installed easily in a straight (portrait) format to increase space in the installation plant.

- Integrated with powerful high-end functionality: can work with archives and VB scripts. Moreover, we can read internet pages and PDF directly in HMI.

2. widescreen displays in a format
    - 50% bigger visualization working area than a normal or conventional display. Intricate working screens can be displayed clearly and divided into segments to monitor and control application.
    - The resolution screen is high and the wide-angle view is very convenient to read and monitor the process.
    - The LED brightness can be easily dimmed and it will easily adapt every lighting condition. This increases the lifetime of the display and saves energy.
3. Application and data protection
    - Backup is automatically done and the device is able to quickly restore the crashed data.
    - In case of power failure, the data is automatically stored in the internal memory of the HMI. Later it can be restored and reused.
    - No additional hardware costs – No external power supply needed.
4. Extremely convenient during the commissioning process
    - Every possible data and setting are stored on the system's memory and it is automatically updated during each process.
    - Budget standard cables are used to download the project data.
    - Commissioning can be done easily and all Ethernet settings configured by the commissioning process.
    - Data transfer can be done easily in the HMI. We can use a flash disk or solid-state drive to transfer the data to any system. and the diagnostics process are done easy by the Simatic controller. We can check the diagnostic report directly in the panel.
    - No external device will be needed to diagnose hardware error
    - We can save time by identifying errors in a very short span of time.
5. It can be easily operated
    - Since it is a mobile keyboard format, entries can be done fast and easy manner.
    - Every key is attached with backlights. Every time a key is pressed, there will be an indication and it helps operators to guide and control the process.
    - There will be tactical feedback, so every time a key is pressed, we can feel the vibration feedback and the key can be operated while wearing gloves.
6. Multiple interfaces with screen and a low hardware cost

- Integrating network structure of PROFIBUS and PROFINET with typical interfaces. According to device sizes and prices, PROFINET ports may increase or decrease.
- There will be pre-installed media player is in the system and that can be used to read the audio/video files.
- External device connections are easy established through the various ports present in the device. we can interface peripheral devices like USB flash drives, mouse, keyboard and printers.

7.2 SIMATIC WinCC in the TIA Portal

It is an HMI software integrated or combined with TIA Portal, it guarantees supreme project efficiency and by integrating all the HMI Panels up, it turns into one big SCADA systems. The software offers ready-to-use objects, reusable faceplates, and intelligent tools, and permits the implementation of multilingual projects. SIMATIC WinCC V11 available in different versions of the TIA Portal, the price and performance are based on the version of TIA Portal which we prefer. The software of the TIA Portal is built easy type, we can import, export or access in all the versions and it takes a small time to import or export one version model into another. Existing old projects can be used in the future versions, there is an option called migrate, this option helps to convert the particular data according to TIA Portal versions. While importing, exporting or migrating the project, all the data are protected and stored in the respective systems or devices. (SIEMENS[13], 2018)

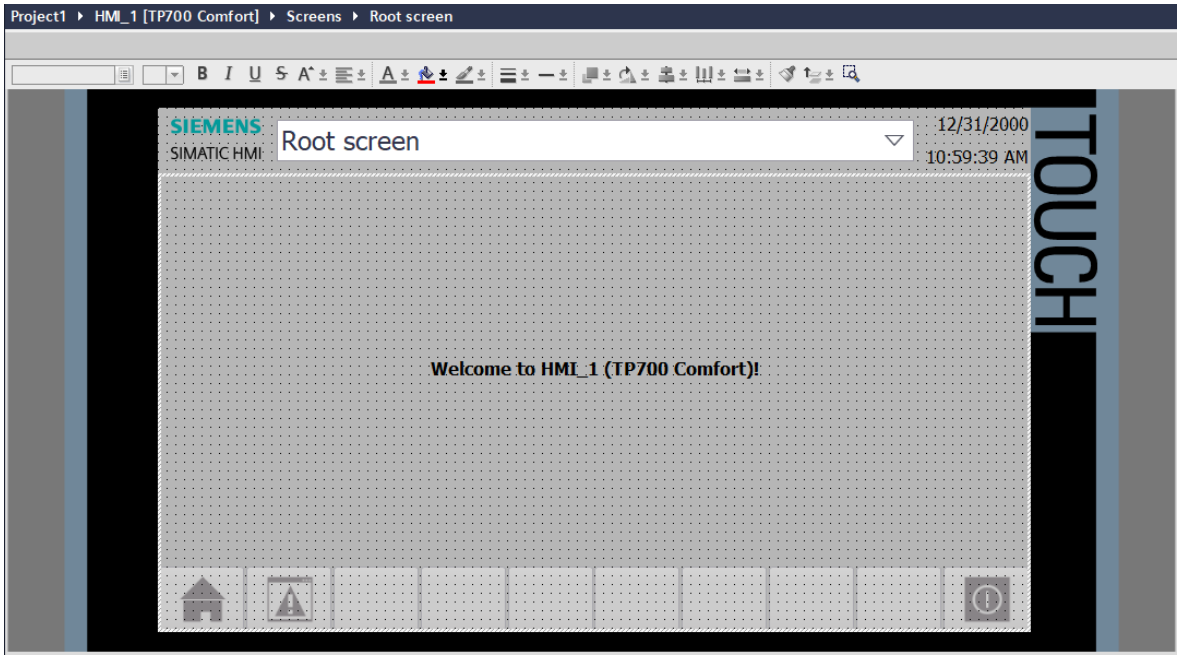


Fig. 39 Project window HMI (in TIA Portal)

7.2.1 Advantages (SIEMENS[8], 2018)

1. Project handling
    - Device-independent configuration data can be used on a variety of target systems without the need for conversion.
    - Project data such as project texts, alarm classes, etc., are able to work c in the TIA Portal and it will be used in various devices for storing and monitoring purposes.
    - A monitoring screen or window is available in the HMI configuration for easily and speedily create a basic structure of a visualization. But it all depends upon current devices that we are working on
2. Editor screen of the HMI is available with so many predefined options and instruction. It will be efficient and visualization can be done in a very fast manner.
    - Screen objects interconnected via Drag & Drop, e.g. we can create and attach tags for every template and all the input/output fields.
    - All the tags can be created in HMI tags and we can drag and drop any tag to any template.
    - screen functions and templates are predefined and the definition of the templates are available in the help.
3. The data management system of HMI is object based and editing options make this interface user-friendly.
    - Logs and alarm configurations are based on the HMI tags. There will be a tab for these options, we can attach respective tags and monitor the situation in HMI screens.
    - By using interfacing option “Cross-reference list”, we can select and access all objects and edit the respective object.
4. Predefined libraries and user-defined configuration objects are available in HMI.
    - All the objects created on the screen from the library are easily stored and can be reused for other projects.
    - In the scope of delivery, a huge number of dynamic and scalable screen objects are available.
5. Commissioning and test backing are available in HMI.
    - In engineering PC, simulation of HMI projects can be done
    - Jump to error cause based on alarm messages in the compiler
6. Migration of existing HMI projects
7. Complete data transfer in projects from WinCC flexible

7.3 TP700 Comfort PLC connection

The physical connection is available for both PLC and an HMI station to exchange data between themselves. A physical connection such as an Ethernet cable, or an RS-485 connection available along with the adequate ports, which helps to establish communication between both PLC and HMI. This is a common protocol that exists on Siemens PLC's and HMI's. Ethernet cable is used for PROFINET communication and RS-485 cable is used for PROFIBUS communication. These are protocols that will help bind the communication interface between PLC and HMI. (SIEMENS[8], 2018)

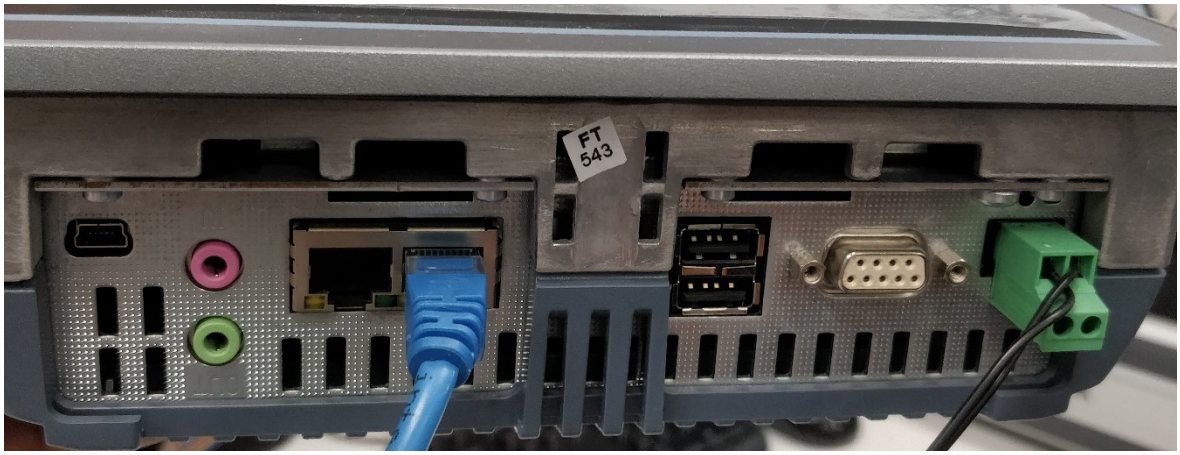


Fig. 40 Communication pins between PLC and HMI

While programming the PLC, the CPU registers control the process, signal alarm states and collect the system status. This is one of the basic parts of the PLC program. Developing the HMI program, in each screen, creating a graphical interface that reflects and send commands to the PLC program through the configured protocol. This is done because, all the tags and registers are linked to the address of PLC so that every alarm display, status icon and input/output field you generate is monitored via HMI screens.

So basically, when the PLC tags are connected in HMI, HMI automatically generates a list of tags related to the PLC tag addresses. Every time a tag is active, CPU reads and activates a particular tag. This activation of a tag is achieved through the communication ports between PLC and HMI. Once the CPU activates a particular tag, it starts to store everything in the memory.

8 BASIC VISUALIZATION OF THERMAL SYSTEM

The goal of visualizing thermal system is to Monitor and control the important parameter of the laboratory model such as

- Monitoring voltage of fan and bulb
- Monitoring of real-time temperature
- Tuning the controller by using the desired value
- Data saving for analysis and future purposes.

These designs are done in TP700 comfort panel using TIA Portal V15. The basic idea of HMI visualization is that the operators can easily access the plant and work with the concerned system in an easy manner. To observe the thermal system visualization, five screens are created in HMI. There is an interconnection button between each screen, which allows us to toggle between all the real-time processes. Process status is provided in the first 2 screens, by using this we will be able to identify which process is working and we would be able to save that particular process data. We can start and stop the process of the system and also save real-time data. Later, the stored data can be used to plot graphs. The process status is created to let the user know if the system is running in an open or closed loop and the corresponding indicator will light up depending on the loop that is being executed. Each button has an indication, when the button is pressed, the colour of the button will change and it will let the users know the concerned button is pressed. It is a basic feedback system for better visualization.

8.1 Screen 1 (LOGIN)

In this screen, Login access is created. The main use of this screen is that it provides security for the system so that a random person does not change any vital parameters that may cause severe damage to the system. Only the administrator and the user will be granted access.

The administrator will have the authority to control all the parameters of the system. On the other hand, users can either monitor or ‘Switch ON' or ‘Switch OFF' the system. All the buttons will remain locked unless the operator of the panel enters the correct/corresponding User ID and Password.

Administrator access:

User-name: admin

Password: admin123

User access:

User-name: user

Password: user123

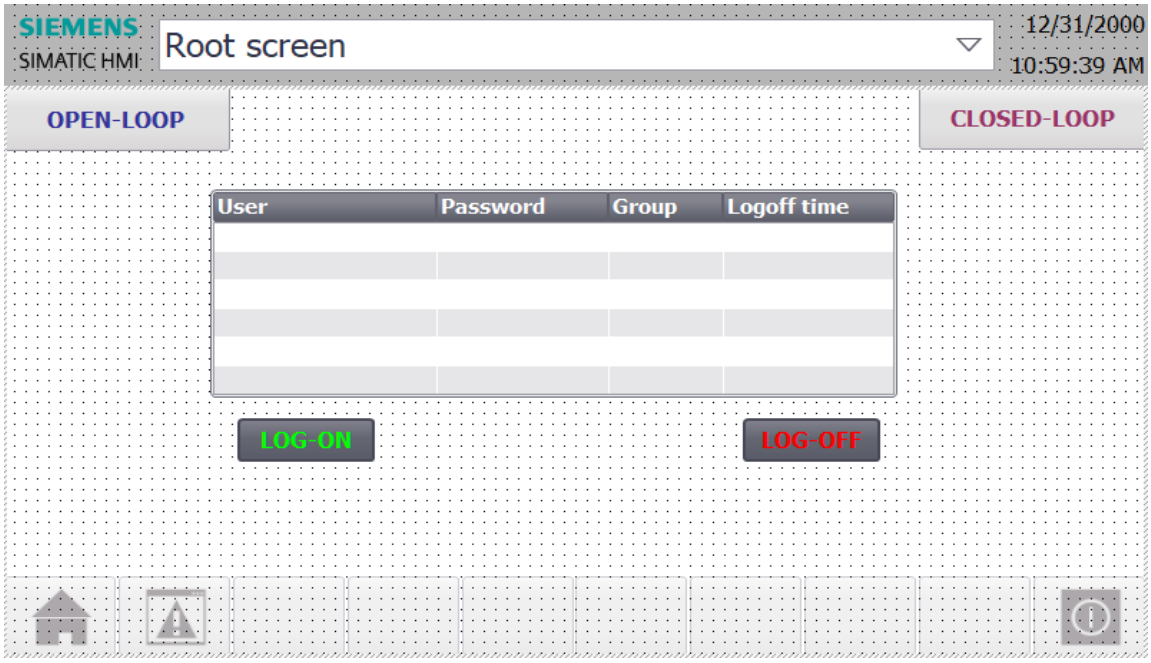


Fig. 41 Login screen

8.2 Screen 2 (OPEN-LOOP CONTROL)

In this screen, open loop control of the thermal system is designed. The voltage value of the bulb and fan is changed to different values to measure the temperature of the system. Since it is an open loop system, there is no feedback or controller to control the system. The fan is acting as a disturbance here so there will be temperature difference in the system. Without any feedback, we cannot control the exact temperature of the system.

In Fig. 42, we can just monitor the temperature and voltage value. And also, the value can be stored for future analysis purposes. This screen is mainly used for system identification purposes. We can feed voltage to bulb and by pressing start logging, we can record temperature and analyse.

Indication

Green colour represents the process is running - start bulb, start the fan and start logging.

Red colour represents the process is at the stop - stop bulb and stop logging.

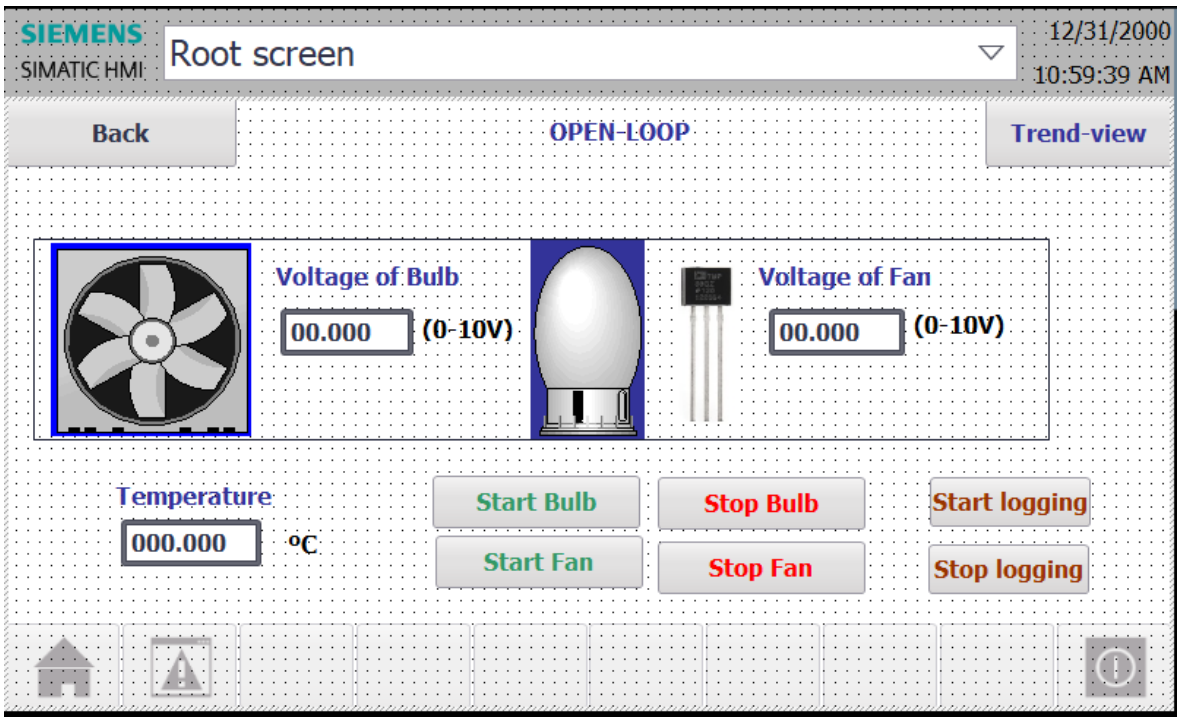


Fig. 42 Open-loop

8.3 Screen 3 (CLOSED -LOOP CONTROL)

In this screen, Closed loop control of the thermal system is designed. It has both controller and feedback in the system. Once the setpoint or desired value is set, the system starts to calculate the error difference between setpoint and output and the error value will be sent into the controller. Then the controller starts to adjust the parameter of the system and controls the system in a smooth manner. The feedback of the system is taken from the temperature sensor TMP36, which is placed next to the bulb. The disturbance from the fan may cause temperature variation in the system. This problem can be controlled by the PID controller by increasing the voltage of the bulb.

The thermal system is controlled by tuning the control parameters and desired value or setpoint of the system. We can see the real-time temperature value and it can be stored for future analysis purposes. The process status is created to let the user know if the system is running in an open or closed loop and the corresponding indicator will light up depending on the loop that is being executed.

Indication

Green colour represents the process is running - start PID and start logging.

Red colour represents the process is at the stop - stop PID and stop logging.

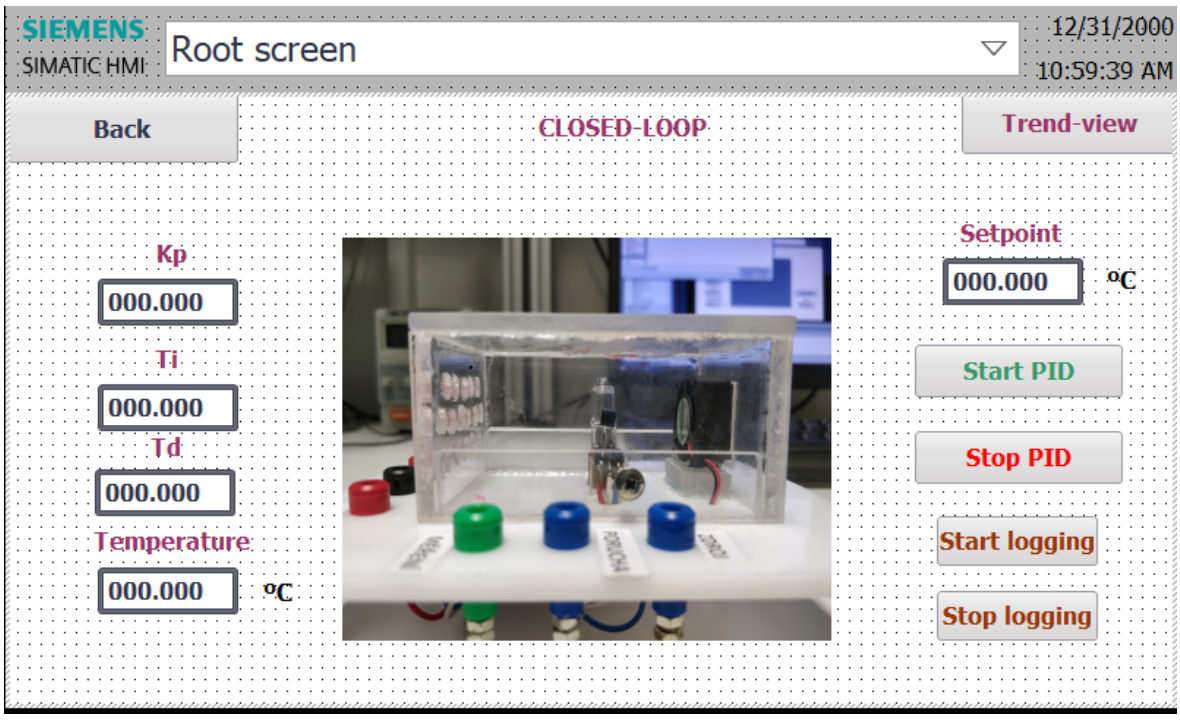


Fig. 43 Closed-loop control

8.4 Screen 4 and 5 (TRACES for open loop and closed loop)

In these screens, traces for both open loop and the closed loop is created. Traces is a real-time graph that helps to monitor the temperature and voltage difference of the system in an instant time.

Traces starts recording automatically when there is a change in the values of tags. We can control the trace by using play, pause, forward, reverse, zoom in and zoom out buttons. By changing these screens, we can monitor both open loop and closed graphs in a single trace screen.

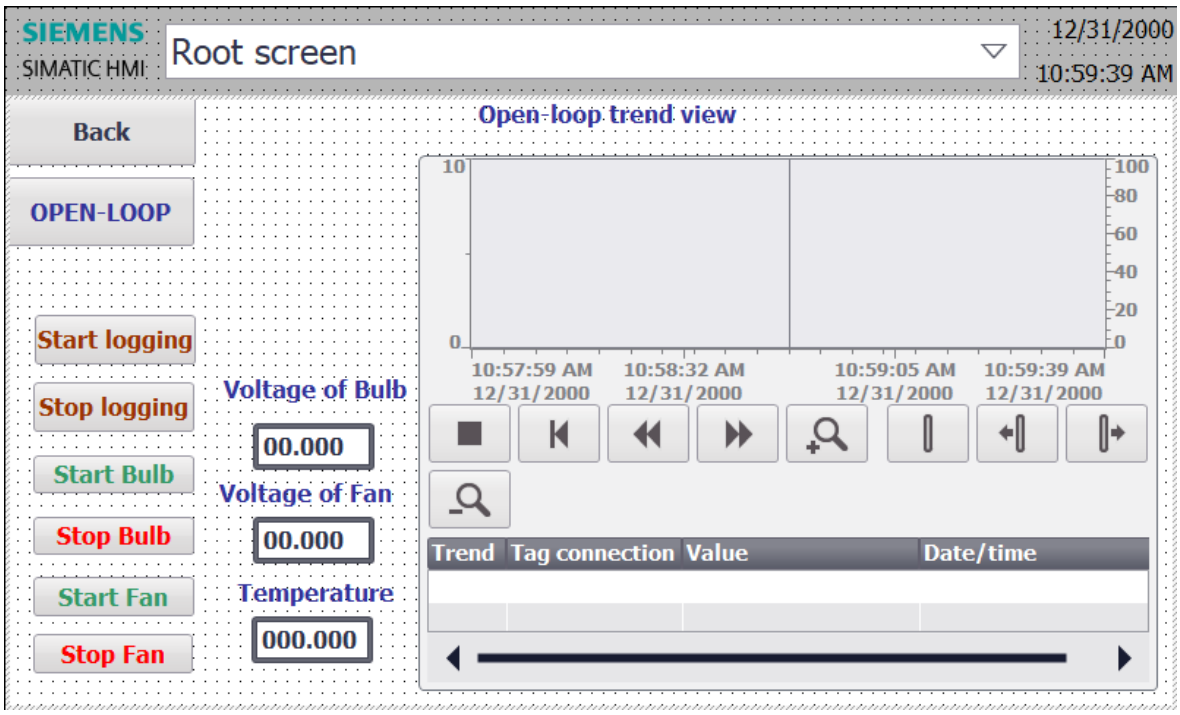


Fig. 44 Traces for open-loop

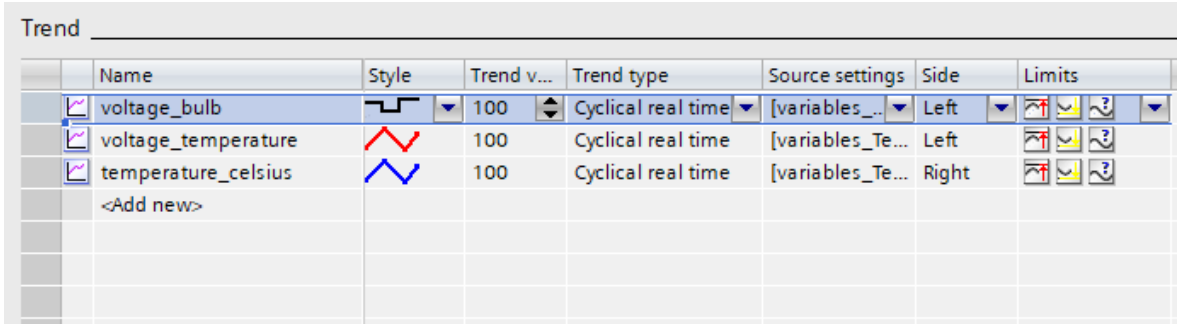


Fig. 45 Parameters for open-loop trace

In the screen of open loop trace, we can set the voltage of both fan and bulb and we can monitor the trace. In traces, I attached some tags like the temperature of the box, the voltage of the bulb and the voltage of the output temperature for monitoring purposes. In open loop trend view screen, we can switch back to both open loop screen and login screen. All these data are stored as a CSV file and analysed in MATLAB

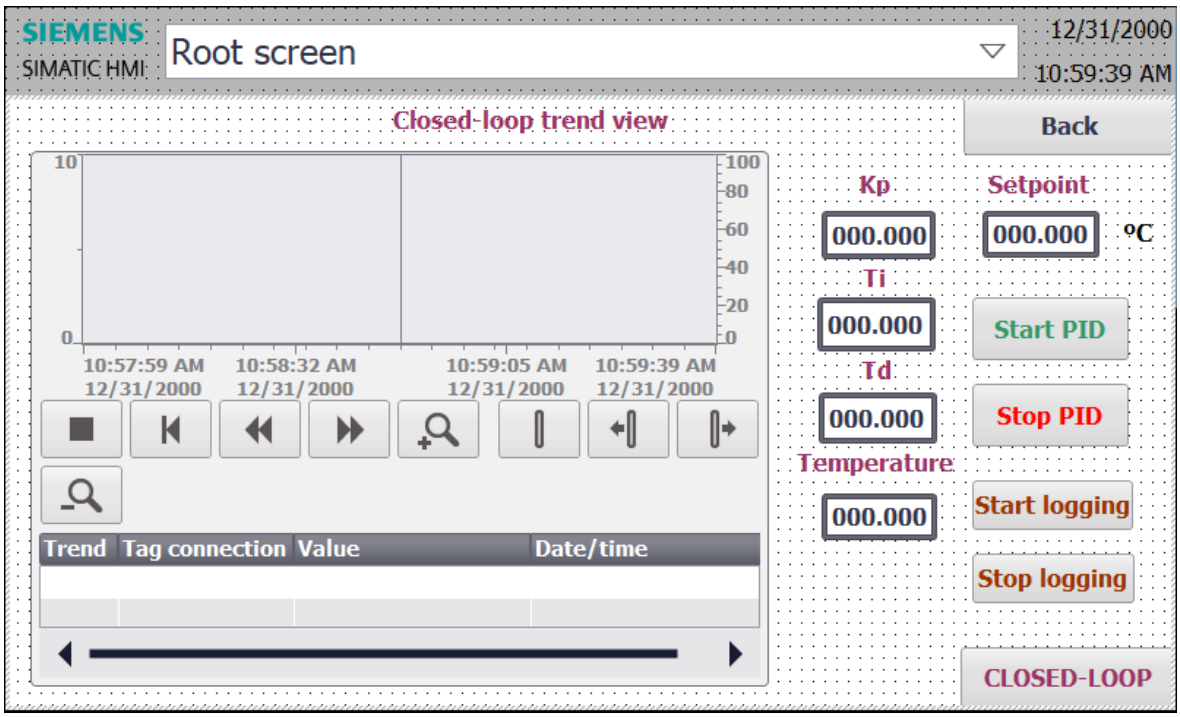


Fig. 46 Traces for closed-loop

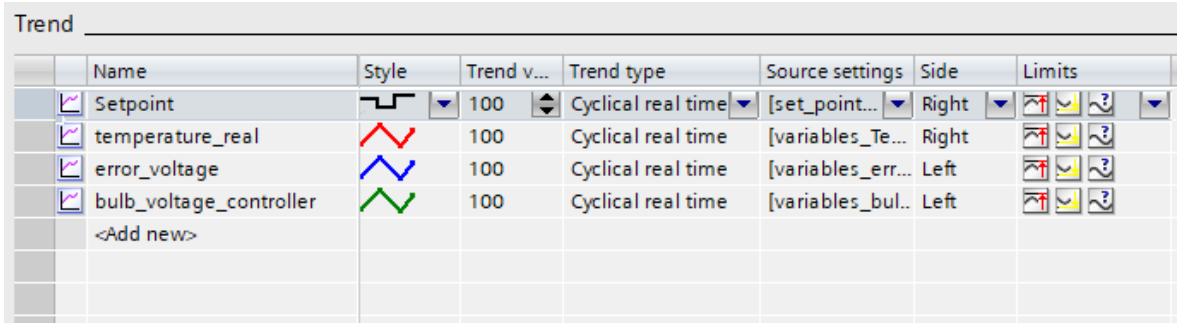


Fig. 47 Parameters for closed-loop trace

In the screen of closed loop trace, we can set the value of K_P, T_I, and T_D and we can monitor the temperature and trace. Similarly, like open loop traces, I attached some tags like the temperature of the box, the voltage of the bulb in controller, error voltage of the controller and setpoint of the closed-loop system for monitoring purposes in closed loop trace. In closed loop trend view screen, we can also switch back to both closed loop screen and login screen. All these data are stored as a CSV file and later used for analysing purposes.

9 DATA LOGGING

Data logging is basically storing the data of particular tags that are connected both PLC and HMI. In the previous chapters, we saw the trace of measuring data and we stored some data for analysis purposes. The saved data is analysed and plotted by using MATLAB. The main task of this chapter is

- Data logging set-up in TIA Portal.
- Importing CSV data file into MATLAB.
- Analysing and plotting graph in MATLAB.

9.1 Data logging set-up in TIA Portal

In TIA Portal V15, WinCC V11 is already integrated with the software. By using WINCC, I created my entire visualization system. In, HMI TP 700 comfort, there is an option called "Historical data", this option will allow us to store the data of tags into the HMI system as two formats either CSV or TXT file.

This historical data helps the system to manage both the data log and alarm log. The data log is storing the data of a particular tag and an alarm tag is helps to indicate the system when the process is wrong or exceeds the certain condition etc, both data log and alarm log runs with the help of logging tag. Logging tag defines the process according to the tags attached to it and it also defines the condition in case of alarm purpose.

In my project, I want to store data for both open loop and closed loop. So, I created data logs for both open loop and closed loop. In Fig. 48 we can see the logging tags of the open-loop process. I wanted to measure input voltage, the temperature in Celsius, temperature as integer values and temperature voltage in open loop and in a closed loop, I wanted measure voltage of the controller, error voltage, set point and temperature in Celsius. This data is stored as CSV and path is mentioned. We can choose our data records and "log handling at the restart" help to clear the log and to store the new updated value into the CSV file every time when the system restarts. Logging cycle of tags can be set in logging tags.

Data logs

...	Name	Storage location	Data records ...	Path	Logging method
	closed_loop	CSV file (ASCII)	5000	\Storage Card USB\traces	Circular log
	open_loop	CSV file (ASCII)	5000	\Storage Card USB\traces	Circular log
	<Add new>				

Logging tags

...	Name	Process tag	Acquisition mode	Logging cycle	High limit
	open_loop_voltage	variables_bulb_open_loop_vol...	Cyclic	1 s	
	Temp_celsius	variables_Temp_celsius	Cyclic	1 s	
	temperature_integer	temp_analog	Cyclic	1 s	
	temperature_voltage	variables_Temp_voltage	Cyclic	1 s	
	<Add new>				

Fig. 48 Data log for open-loop

Data logs

...	Name	Storage location	Data records ...	Path	Logging method
	closed_loop	CSV file (ASCII)	5000	\Storage Card USB\traces	Circular log
	open_loop	CSV file (ASCII)	5000	\Storage Card USB\traces	Circular log
	<Add new>				

Logging tags

...	Name	Process tag	Acquisition mode	Logging cycle	High limit
	bulb_voltage_controller	variables_bulb_output_voltage	Cyclic	1 s	
	error_voltage	variables_error_voltage	Cyclic	1 s	
	set_point	set_point_temp	Cyclic	1 s	
	Temperature_celsius	Temp_move	Cyclic	1 s	
	<Add new>				

Fig. 49 Data log for closed-loop

9.2 Importing CSV data file into MATLAB

After opening MATLAB, in the home tab, there will be an option called import. By pressing import, we select the path of data and import the CSV data file. Once the data is opened in the import section, we have to select the column vector as I have shown in Fig. 51. And we have to use delimiters and separators to split the data as I have shown in Fig. 52 and Fig. 53. Once the data is separated, we have to specify names for two columns. We should name the “varname” as “names” and “varvalues as "values". After changing the names, select the data from the 2nd row as I shown in Fig. 54 and press import. The imported values are stored in the workspace. We have to sort the imported data to get proper results. Once everything is done, we can plot the data and check the results.

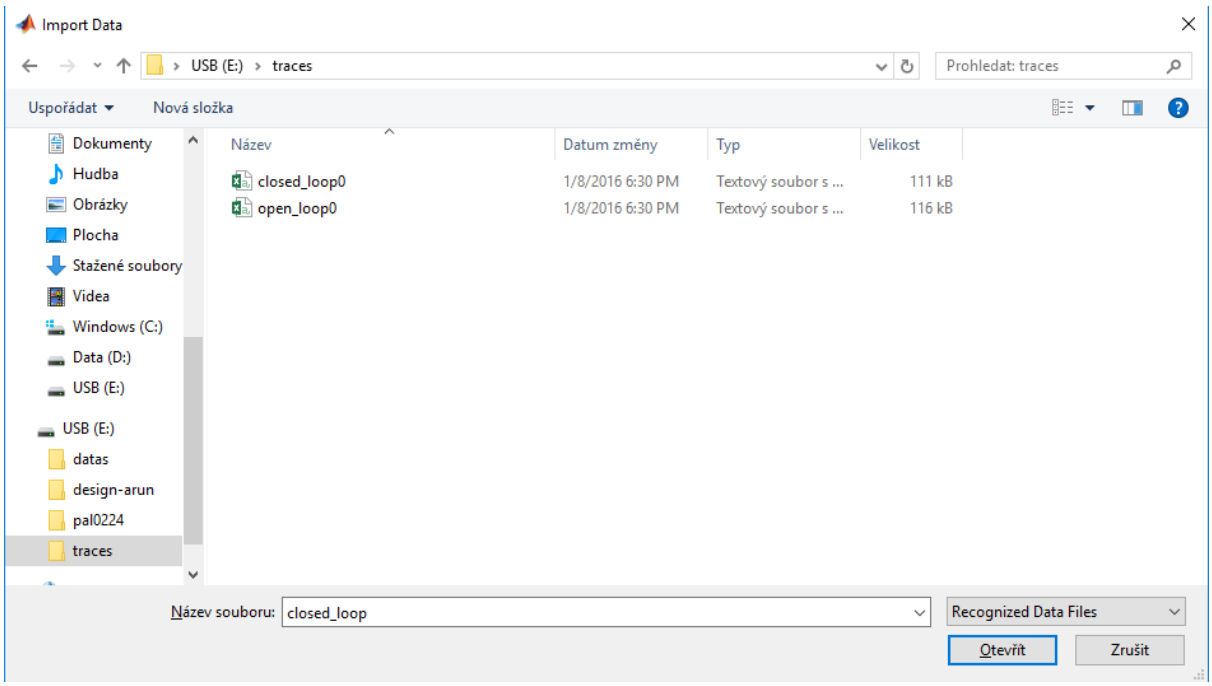


Fig. 50 Path of the CSV file

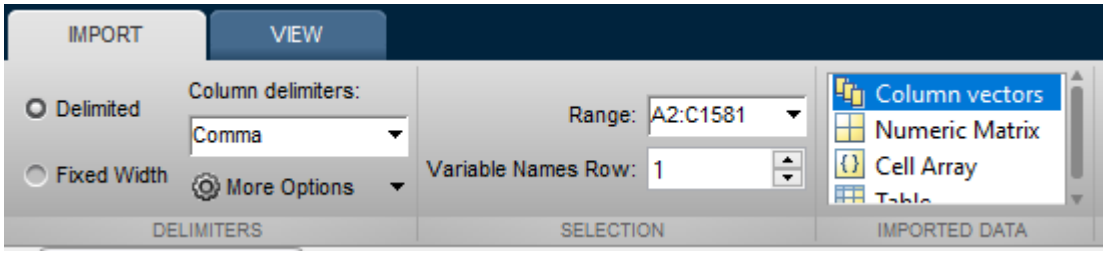


Fig. 51 Selecting column vector in the import section

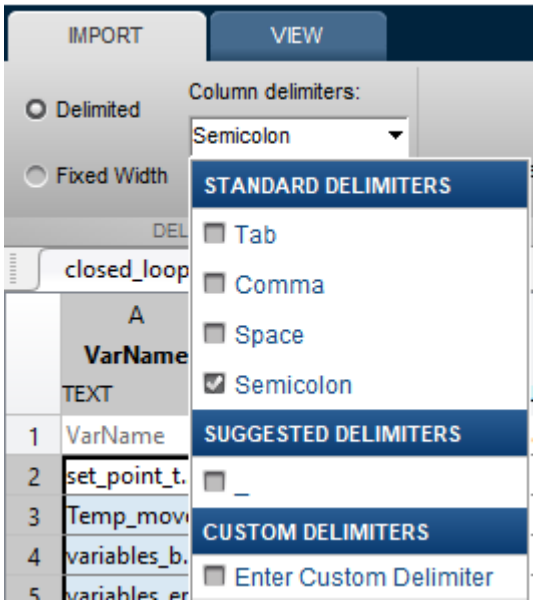


Fig. 52 Selecting semicolon delimiter (import section)

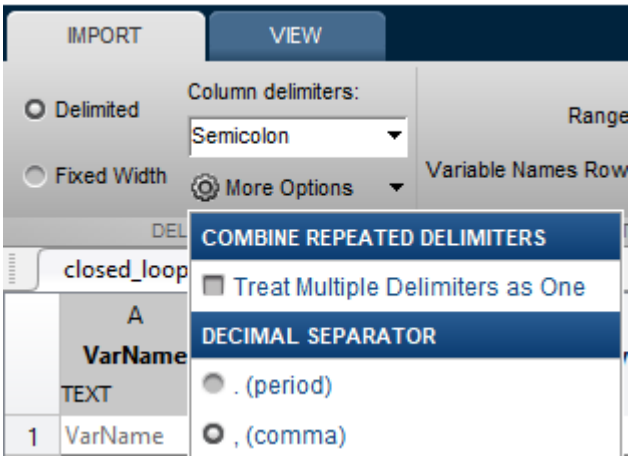
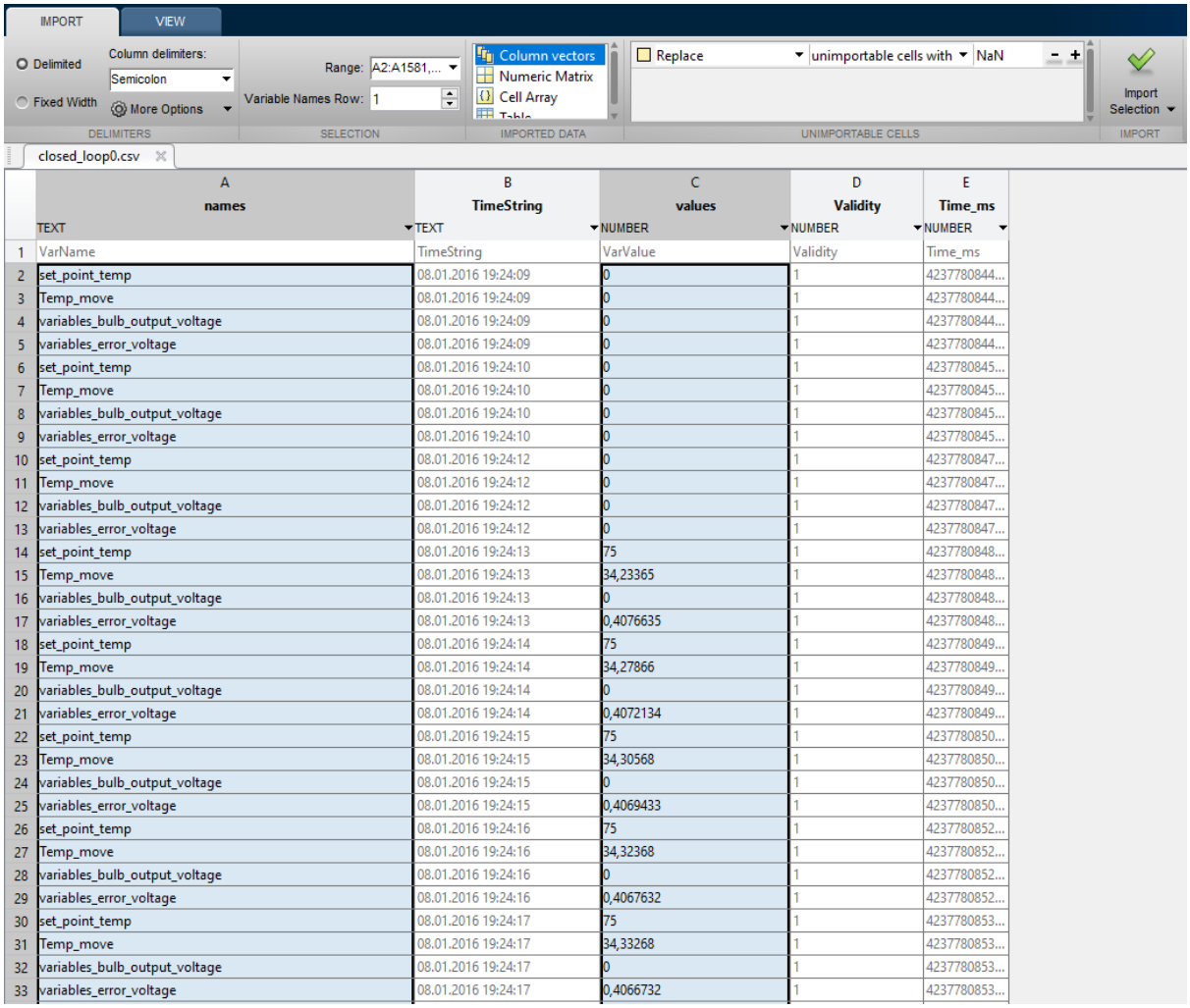


Fig. 53 Selecting comma separator in (import section)



	A names	B TimeString	C values	D Validity	E Time_ms
	TEXT	TEXT	NUMBER	NUMBER	NUMBER
1	VarName	TimeString	VarValue	Validity	Time_ms
2	set_point_temp	08.01.2016 19:24:09	0	1	4237780844...
3	Temp_move	08.01.2016 19:24:09	0	1	4237780844...
4	variables_bulb_output_voltage	08.01.2016 19:24:09	0	1	4237780844...
5	variables_error_voltage	08.01.2016 19:24:09	0	1	4237780844...
6	set_point_temp	08.01.2016 19:24:10	0	1	4237780845...
7	Temp_move	08.01.2016 19:24:10	0	1	4237780845...
8	variables_bulb_output_voltage	08.01.2016 19:24:10	0	1	4237780845...
9	variables_error_voltage	08.01.2016 19:24:10	0	1	4237780845...
10	set_point_temp	08.01.2016 19:24:12	0	1	4237780847...
11	Temp_move	08.01.2016 19:24:12	0	1	4237780847...
12	variables_bulb_output_voltage	08.01.2016 19:24:12	0	1	4237780847...
13	variables_error_voltage	08.01.2016 19:24:12	0	1	4237780847...
14	set_point_temp	08.01.2016 19:24:13	75	1	4237780848...
15	Temp_move	08.01.2016 19:24:13	34,23365	1	4237780848...
16	variables_bulb_output_voltage	08.01.2016 19:24:13	0	1	4237780848...
17	variables_error_voltage	08.01.2016 19:24:13	0,4076635	1	4237780848...
18	set_point_temp	08.01.2016 19:24:14	75	1	4237780849...
19	Temp_move	08.01.2016 19:24:14	34,27866	1	4237780849...
20	variables_bulb_output_voltage	08.01.2016 19:24:14	0	1	4237780849...
21	variables_error_voltage	08.01.2016 19:24:14	0,4072134	1	4237780849...
22	set_point_temp	08.01.2016 19:24:15	75	1	4237780850...
23	Temp_move	08.01.2016 19:24:15	34,30568	1	4237780850...
24	variables_bulb_output_voltage	08.01.2016 19:24:15	0	1	4237780850...
25	variables_error_voltage	08.01.2016 19:24:15	0,4069433	1	4237780850...
26	set_point_temp	08.01.2016 19:24:16	75	1	4237780852...
27	Temp_move	08.01.2016 19:24:16	34,32368	1	4237780852...
28	variables_bulb_output_voltage	08.01.2016 19:24:16	0	1	4237780852...
29	variables_error_voltage	08.01.2016 19:24:16	0,4067632	1	4237780852...
30	set_point_temp	08.01.2016 19:24:17	75	1	4237780853...
31	Temp_move	08.01.2016 19:24:17	34,33268	1	4237780853...
32	variables_bulb_output_voltage	08.01.2016 19:24:17	0	1	4237780853...
33	variables_error_voltage	08.01.2016 19:24:17	0,4066732	1	4237780853...

Fig. 54 Selecting 2 columns data name and import (import section)

9.3 Analysing and plotting graph in MATLAB

In this section, imported data are separated according to names and their respective values. To separate these values and names, I have created an m-file (MATLAB file) for both open and closed loop. Once the data is imported, open the respective m-file and press run option from the editor's tab in MATLAB.

Once the m-file is executed, sorted data are created in the workspace. We can plot all the data and see the results for real-time data that we previously stored as a CSV file. By using this m-file we can sort and plot the data in an easy manner.

Editor - C:\Users\h310\Documents\Automation\pal0224\openloop.m

openloop.m

```
clear open_loop_bulb_voltage
clear temp_celsius
clear temp_voltage
clear temp_analog
clear i
clear x
clear t

clc

open_loop_bulb_voltage = zeros(length(names)/4,1);
temp_celsius = zeros(length(names)/4,1);
temp_voltage = zeros(length(names)/4,1);
temp_analog = zeros(length(names)/4,1);


for i = 1:(length(values)/4)
    for x = 1:4
        switch x
            case 1
                open_loop_bulb_voltage(i,1) = values((i-1)*4 + x,1);
            case 2
                temp_celsius(i,1) = values((i-1)*4 + x,1);
            case 3
                temp_voltage(i,1) = values((i-1)*4 + x,1);
            case 4
                temp_analog(i,1) = values((i-1)*4 + x,1);
        end
    end
end

t = (0:1:(i-1))';
plot(t,temp_voltage,t,open_loop_bulb_voltage,t,temp_celsius);
grid on;
legend('temp voltage','open loop bulb voltage','temp celsius');
```

Fig. 55 Code for separating open-loop data

Editor - C:\Users\h310\Documents\Automation\pal0224\closedloop.m

closedloop.m

```
clear set_point_temp
clear temp_celsius
clear error_voltage
clear controller_output_votage
clear i
clear x
clear t

clc

set_point_temp = zeros(length(names)/4,1);
temp_celsius = zeros(length(names)/4,1);
controller_output_votage = zeros(length(names)/4,1);
error_voltage = zeros(length(names)/4,1);


for i = 1:(length(values)/4)
    for x = 1:4
        switch x
            case 1
                set_point_temp(i,1) = values((i-1)*4 + x,1);
            case 2
                temp_celsius(i,1) = values((i-1)*4 + x,1);
            case 3
             controller_output_votage(i,1) = values((i-1)*4 + x,1);
            case 4
                error_voltage(i,1) = values((i-1)*4 + x,1);
        end
    end
end

t = (0:1:(i-1))';
plot(t,error_voltage,t,controller_output_votage,t,set_point_temp,t,temp_celsius);
grid on;
legend('error voltage','controller output votage','set point temp','temp celsius');
```

Fig. 56 Code for separating closed-loop data

Workspace

Name	Value
i	314
names	*1256x1 cell*
open_loop_bulb_v...	*314x1 double*
t	*314x1 double*
temp_analog	*314x1 double*
temp_celsius	*314x1 double*
temp_voltage	*314x1 double*
values	*1256x1 double*
x	4

Workspace

Name	Value
controller_output_...	*395x1 double*
error_voltage	*395x1 double*
i	395
names	*1580x1 cell*
set_point_temp	*395x1 double*
t	*395x1 double*
temp_celsius	*395x1 double*
values	*1580x1 double*
x	4

Fig. 57 The workspace of open-loop and closed-loop

10 EXPLANATION OF CLOSED LOOP CONTROL ALGORITHM AND RESULTS

The process of a closed control loop system of laboratory model is explained below. In general, elements of the closed-loop system consist of setpoint, error, controller, input, plant, disturbance, output, and feedback.

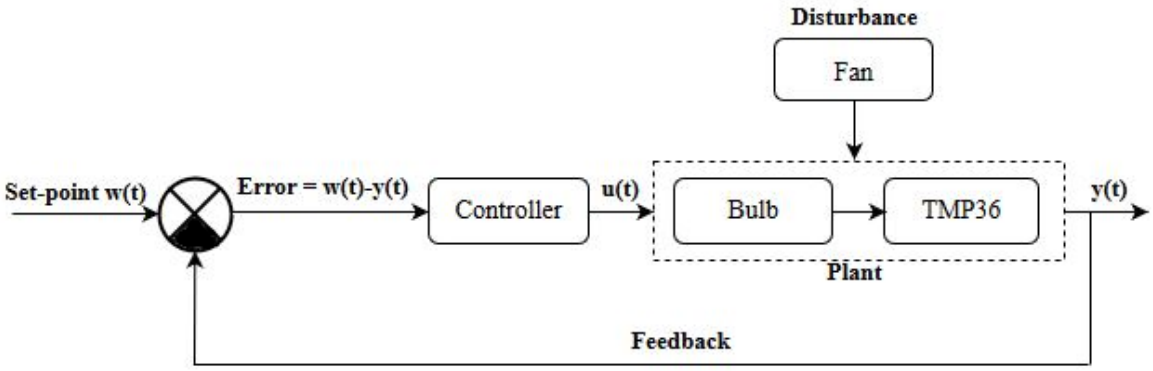


Fig. 58 Closed-loop control system

To design the closed-loop algorithm, I made all the values in one physical element. The physical element I have chosen to create this algorithm is voltage.

The setpoint value will be given in terms of ºC, later the degree Celsius is converted into voltage and fed into the error.

Measured output (y) will be measured by the temperature sensor. The value of the temperature sensor is later converted into voltage and fed into the error as negative feedback.

Error is the difference between set point voltage and output temperature voltage. Once the error is calculated, it will be fed into the controller.

In the controller, control parameters would be able to produce some voltage value of bulb. Later the voltage value of the bulb is recalculated in terms of analog value before feeding it into the plant (u).

Once the integer value is fed into the plant, bulb starts to glow and we can measure and maintain the temperature of the box.

The fan is acting as a disturbance in the system. When the fan is switched on, the temperature will start to reduce. Due to the value changes in the temperature, error starts to rise and the controller will try to increase the voltage of the bulb to maintain the box temperature.

10.1 Open-loop results

The results for an open loop system are shown and the process is explained below. In Fig. 59, we can see that the open loop system is in the stop mode and the voltage of both bulb and fan is zero. Since no process is running, process status indicates everything is in off state and the temperature is at 25ºC.

In Fig. 60, we can see that the voltage of the bulb is set to 4V. By pressing the start button, we can start the bulb and the green light indicates the system is in the running mode.

In Fig. 61, we can see the real-time trends and by pressing start logging the data started to store in the HMI system. This is the design of my open-loop system and it is majorly used for the system identification. By giving voltage to the bulb, we can measure and save the response of the temperature sensor. Later the saved data is plotted in MATLAB and the transfer function is identified.

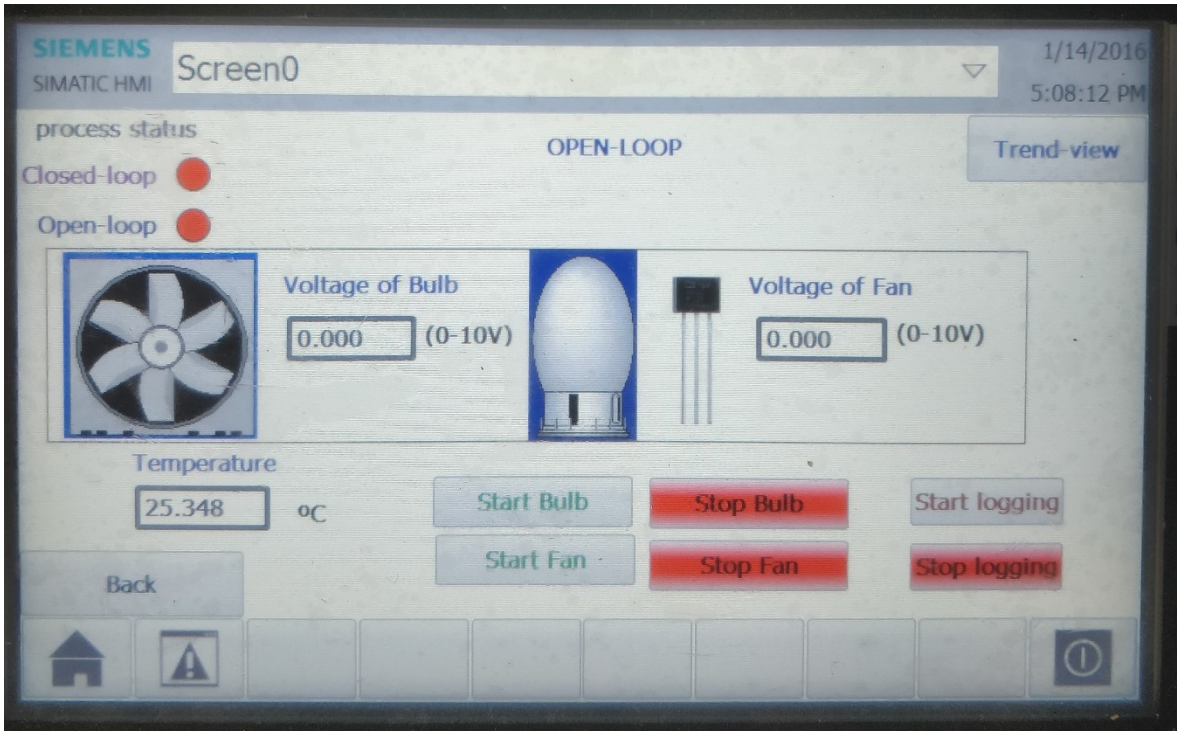


Fig. 59 Live HMI screen of open-loop in stop mode

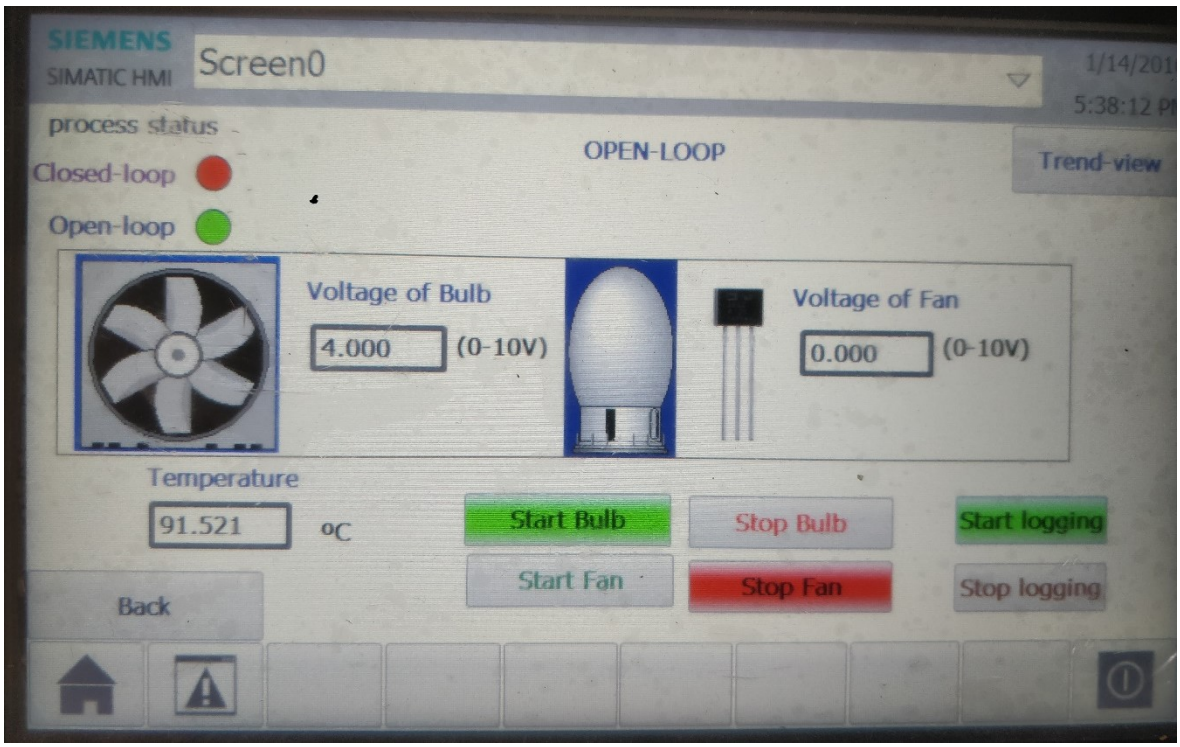


Fig. 60 Live HMI screen of open-loop in start mode

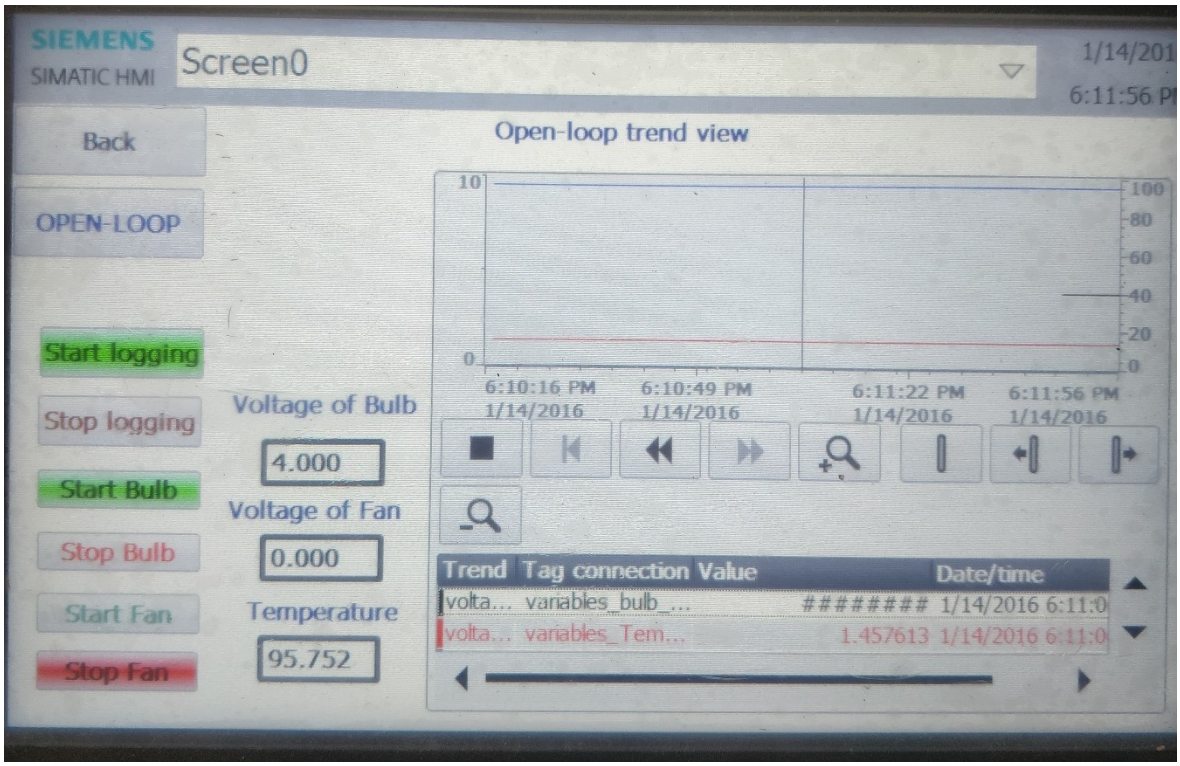


Fig. 61 Live HMI screen of open-loop trend view

10.2 Closed loop results

Results for the closed-loop control system is shown and explained below. In Fig. 62, we can see that the red colour buttons in the HMI screen. So, the closed-loop system is in the stop mode. As I mentioned earlier, this system works properly in the PI controller.

After experimenting for a long time, I would be able to find the right parameters for the controller. It's a PI controller, the value of K_p is 15 and the value of is T_I is 60. These controller values work fine even with disturbance.

Once the parameter and the setpoint are set, by pressing start PID, we can start the system and it starts to control the temperature according to the set point. In Fig. 63, flashing green colour button indicates the system is on mode.

I designed my controller as operator friendly, so the operator can choose P, PI, PD or PID. Even though the controller design is operator friendly, I would suggest the PI controller for this system.

In Fig. 64, setpoint is set to 75ºC. Once the parameter values are set, the controller starts to produce voltage and increases the temperature. The temperature will be maintained with the help of error value before the controller. In the below picture, we can see that the temperature is maintained at 74ºC.

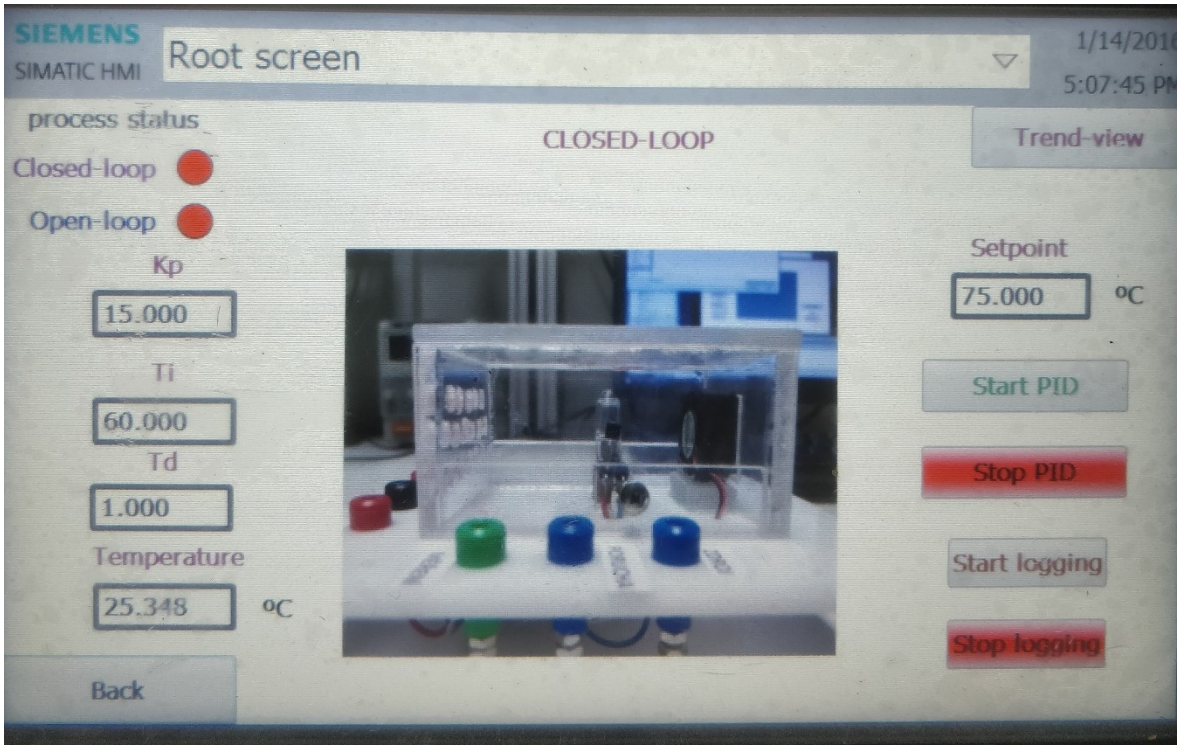


Fig. 62 Live HMI screen of closed-loop in stop mode

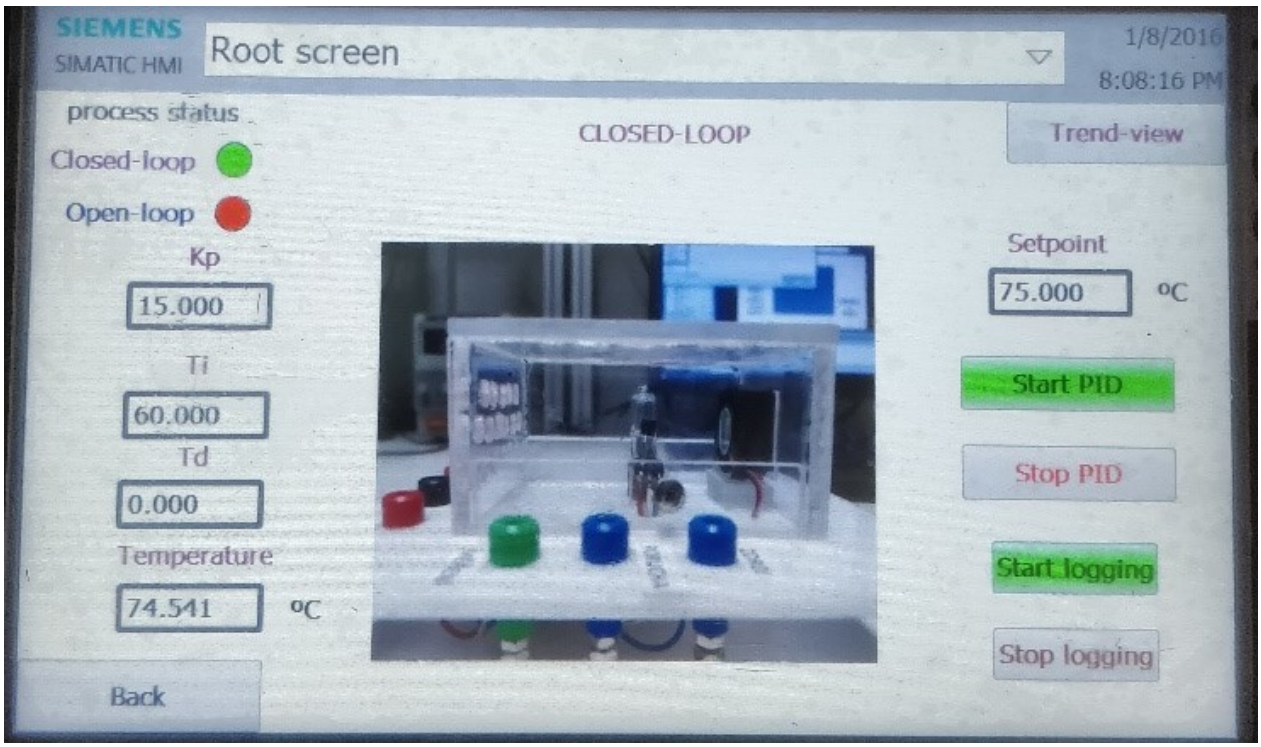


Fig. 63 Live HMI screen in start mode

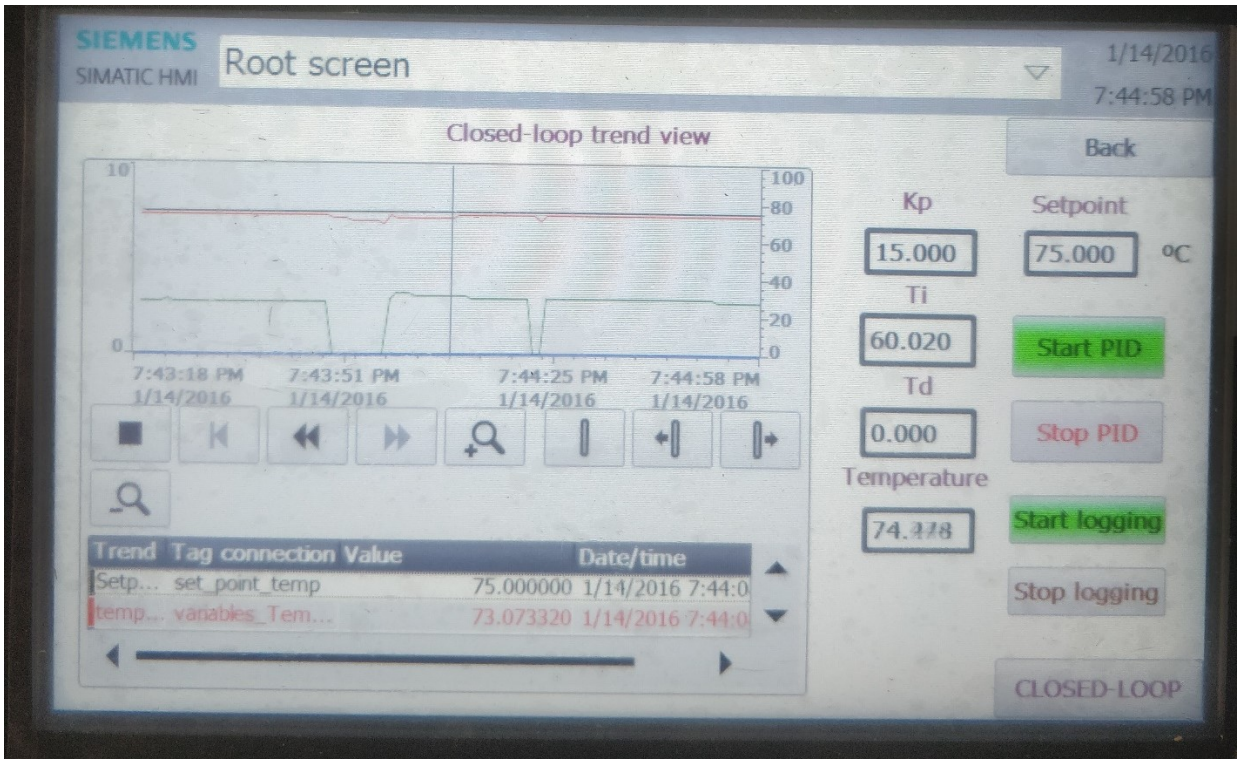


Fig. 64 Live HMI screen of closed-loop trend view

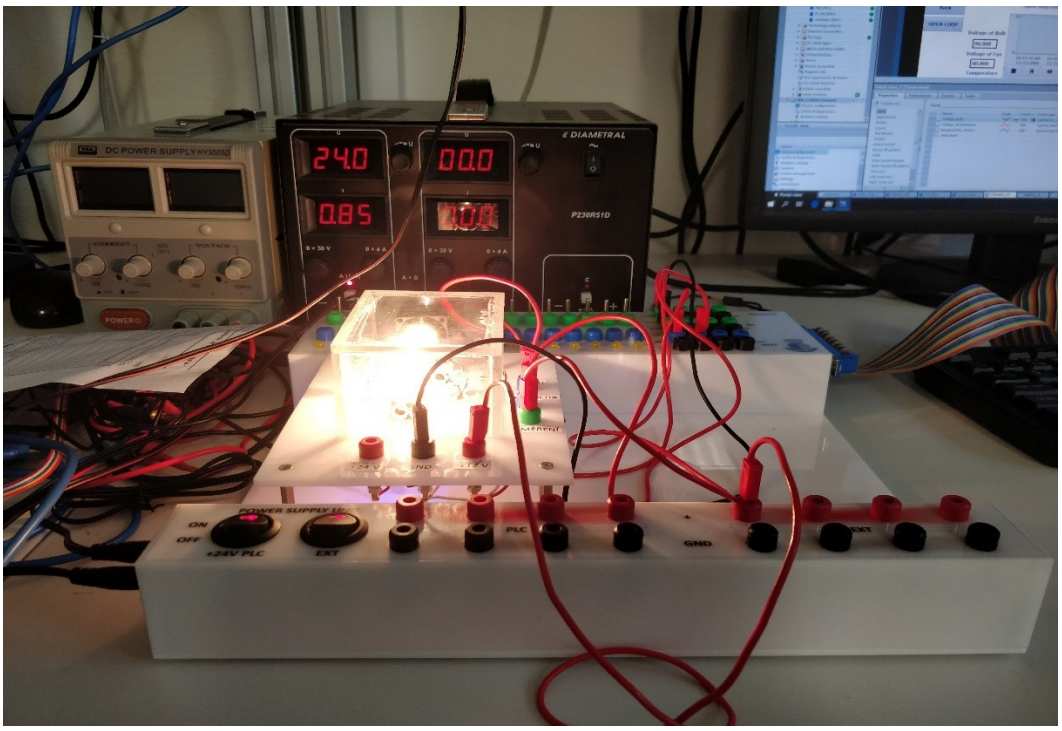


Fig. 65 Laboratory model

Trend view of the closed-loop system and picture of the real-time model are shown above. By clicking start logging, we can save all the values and traces data for future analysis purpose.

10.3 Data logging results

Data logging results are processed by MATLAB open loop and closed loop m-files. Once the imported data is executed, it will automatically sort the data and plot the graphs. Those graph results are shown below.

In Fig. 66, we can see that the bulb voltage is set to 4V and the temperature is automatically increased to almost 90ºC and starts settling at 96ºC. It is an open loop process and it is mainly used for system identification.

In Fig. 67, we can see the analog value of the temperature sensor TMP36 with respective time axis.

In Fig. 68, we can see that the set point is 75ºC and the controller voltage slowly starts to increase, respectively temperature is also increasing. We can see the temperature is controlled at 75ºC with the help of error value and controller voltage. This result represents the closed loop system.

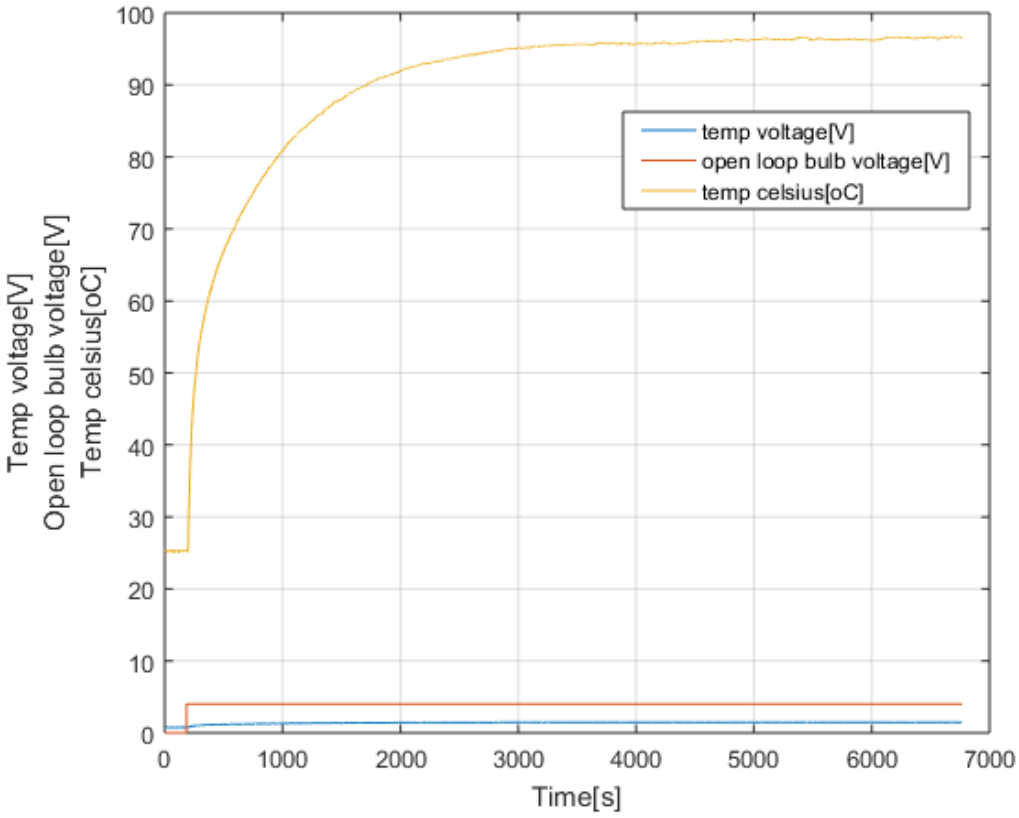


Fig. 66 Data logging results for open-loop process

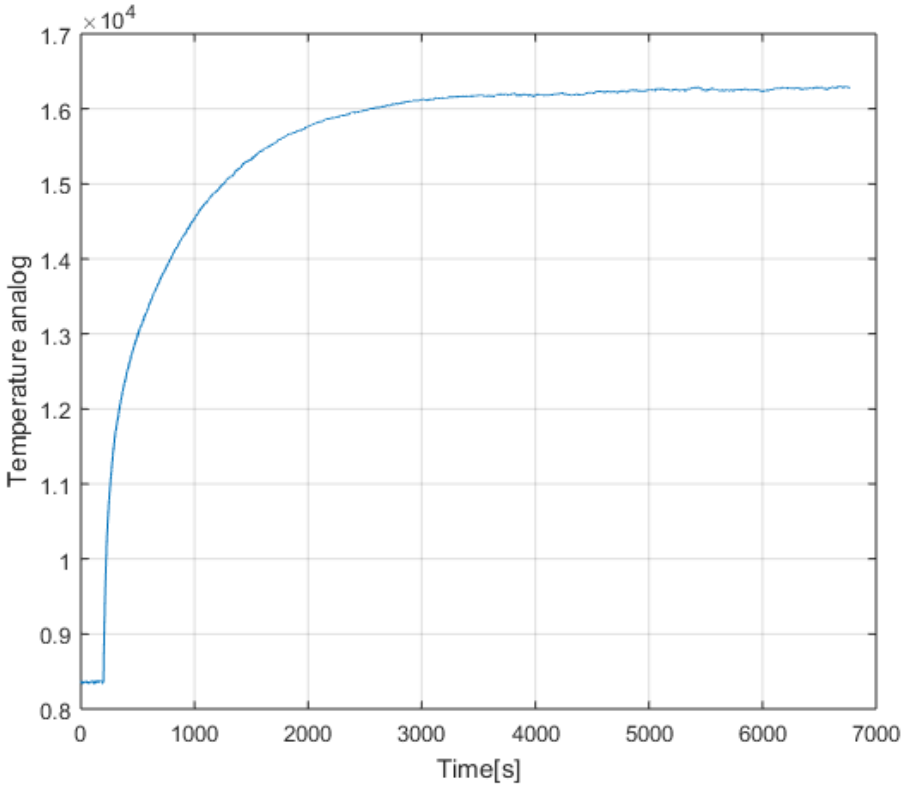


Fig. 67 Data logging results for temperature analog (open-loop)

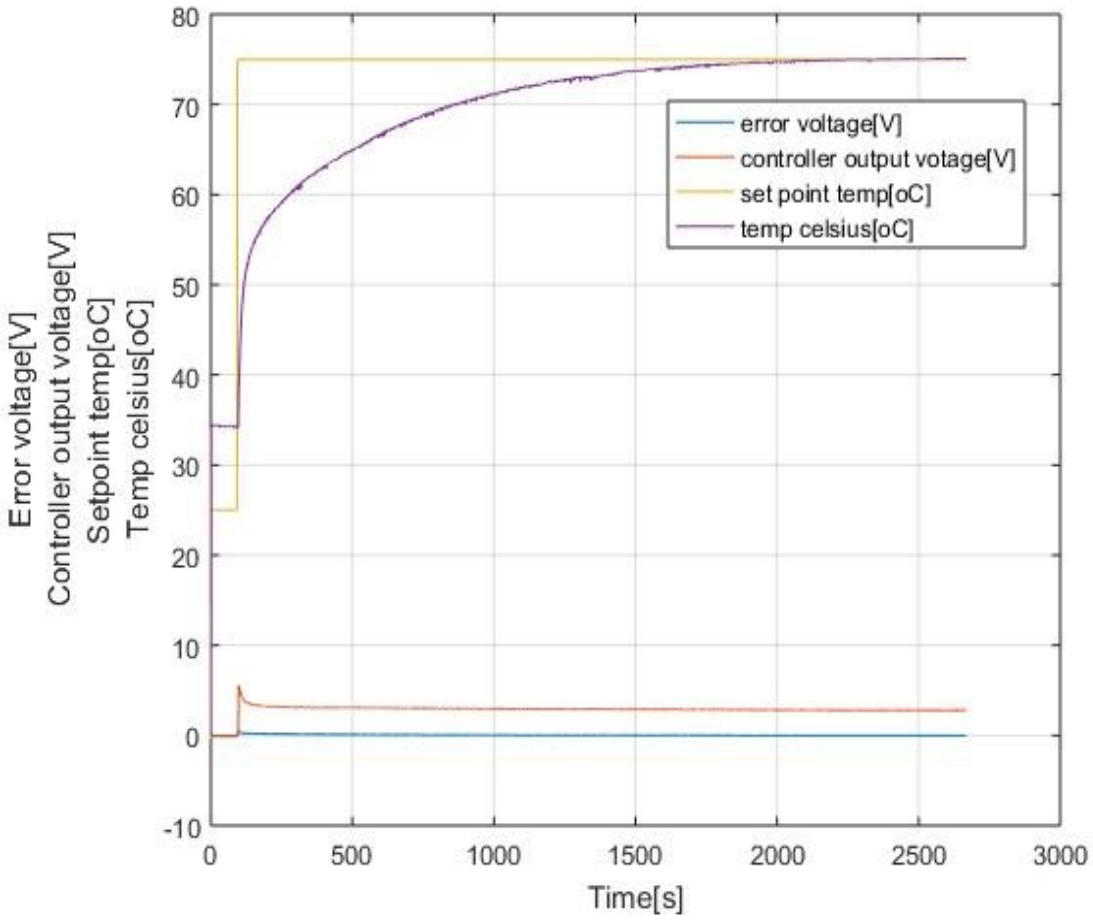


Fig. 68 Data logging results for closed-loop process

CONCLUSION

The description of the Siemens S7-1500 PLC and TP700 comfort HMI panel, software tools and programming techniques were explained in a detailed manner. The electrical circuit for the thermal system was designed and created successfully. The PCB design of the system was completed and the soldering process was done. PCB was attached to the laboratory model and the diagnosis and testing were done. Later, system identification in the open-loop was executed and step response was measured. With the help of step response, the transfer function was identified. By executing the transfer function in MATLAB Simulink, I was able to identify the K_P, T_I and T_D values of the system. After obtaining tuning parameters, PLC control algorithm of the system was created for both open and closed loop system. The control algorithm for both open and closed loop was demonstrated. Also, the design in the HMI panel was done (TP700 Comfort HMI Touch Panel) and connected with real tags of the thermal system. Results for the visualization were explained step by step along with the explanation about data logging process.

In this work, system temperature was controlled in the range of setpoint or pre-set value. By setting the temperature in the set point, the PID controller helps the system to maintain the temperature in terms of error value of the output and setpoint. The PID controller provides output voltage according to the error value. If the system is subjected to outside interference such as disturbance, then the system temperature will be automatically controlled with the help of the controller. After experimenting with the laboratory model for a long time, ideal tuning parameters were found. As mentioned earlier, the suitable controller for this laboratory model is PI. Ideal tuning parameter for PI is $K_p = 15$ and $T_I = 60$. By using this tuning parameter, the box temperature was achieved and controlled successfully. I believe, this work will expect to bring a good application prospect.

This laboratory model may be used for teaching purposes or practice model for the students in the future. By working with a real-time model, students would have a better understanding of the open loop (system identification) and closed-loop control system.

REFERENCES

ANALOG AUTHORS, 2018. *Low voltage temperature sensor*. Dostupné také z: http://www.analog.com/media/en/technical-documentation/data-sheets/TMP35_36_37.pdf

AUTODESK, Inc, 2018. *Temperature sensor tutorial*. Dostupné také z: https://www.instructables.com/id/Temperature-Sensor-Tutorial/

AUTODESK, Inc, 2019. *Eagle software tutorial*. Dostupné také z: https://www.autodesk.com/products/eagle/blog/schematic-basics-part-1/

GOWRI shankar, , 2008. Control of Boiler operation using PLC-SACDA. In: *Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II*. Hong kong, s. 6. DOI: ISBN: 978-988-17012-1-3.

MATHWORKS, 2019. Videos and Web Seminars. *PID Controller Tuning in MATLAB Simulink*. Dostupné také z: https://www.mathworks.com/videos/getting-started-with-simulink-part-4-tuning-a-pid-controller-1508444927396.html

MRÁZEK, Lukáš, 2016. *Design of Educational Tasks for Model with PLC Controller*. VŠB – Technical University of Ostrava. Bachelor Thesis. VŠB – Technical University of Ostrava.

PFEIFFER, Bernd-Markus, 1999. TOWARDS „PLUG&CONTROL“:SELFTUNING TEMPERATURE CONTROLLER FOR PLC. In: *European Control Conference*. Karlsruhe, Germany, s. 6. DOI: 10.23919/ECC.1999.7099688. Dostupné také z: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7099688

PLC MANUAL, 2018. *PLC MANUAL*. Dostupné také z: http://www.plcmanual.com/plc-programming

SIEMENS[1], 2018. Analog Input Module S7-1500. *Manual*. Dostupné také z: https://cache.industry.siemens.com/dl/files/205/59193205/att_112065/v1/s71500_ai_8xu_i_rtd_tc_st_manual_en-US_en-US.pdf

SIEMENS[10], 2018. *HMI TP700 Comfort System Overview*. Dostupné také z: https://w3.siemens.com/mcms/human-machine-interface/en/operator-devices/advanced-hmi-panel-based/comfort-panels/system-overview/Pages/Default.aspx

SIEMENS[11], 2017. Programming Guidelines for S7-1500. *Manual*. Dostupné také z: http://www1.siemens.cz/ad/current/content/data_files/automatizacni_systemy/mikrosystemy/simatic_s71200/programming-guideline-for-s71200-s71500_2014-09_en.pdf

SIEMENS[12], 2018. S7-1500 Getting Started Manual. *Manual*. Dostupné také z: https://www.automation.siemens.com/salesmaterial-as/interactive-manuals/getting-started_simatic-s7-1500/documents/EN/software_complete_en.pdf

SIEMENS[13], 2018. WINCC Software Manual. *Manual*. Dostupné také z: https://cache.industry.siemens.com/dl/files/270/1145270/att_36311/v1/Confm1_e.pdf

SIEMENS[14], 2018. WINCC System Description Manual. *Manual*. Dostupné také z: https://w5.siemens.com/italy/web/AD/ProdottieSoluzioni/Sistemiautomazionenew/SIMATICHMI/SimaticHMIsoftware/Documents/wincc/WinCC%20V7%20Descizione%20tecnica%20del%20sistema%20-%20%20Settembre%202008%20(ENG).pdf

SIEMENS[2], 2018. Analog Output Module. *Manual*. Dostupné také z: https://cache.industry.siemens.com/dl/files/850/59191850/att_63218/v1/s71500_aq_4xu_i_st_manual_en-US_en-US.pdf

SIEMENS[3], 2018. Automation support. *Siemens*. Dostupné také z: http://support.automation.siemens.com/ww/llisapi.dll?func=cslib.csinfo&lang=en&objid=6es75163an000ab0&caller=view.

SIEMENS[4], 2018. Communication Module S7-1500. *Manual*. Dostupné také z: https://cache.industry.siemens.com/dl/files/160/59057160/att_2515/v1/s71500_cm_ptp_rs232hf_manual_en-US_en-US.pdf

SIEMENS[5], 2018. DIgital Input Module. *Manual*. Dostupné také z: https://support.industry.siemens.com/cs/document/59192896/simatic-s7-1500-et-200mp-digital-input-module-di-32x24vdc-hf?dti=0&pnid=13715&lc=en-UY

SIEMENS[6], 2018. Digital Output Module. *Manual*. Dostupné také z: https://support.industry.siemens.com/cs/document/59193400/simatic-s7-1500-et-200mp-digital-output-module-dq-32x24vdc-0-5a-st?dti=0&pnid=13716&lc=en-CZ

SIEMENS[7], 2018. *HMI Comfort PanelsTP700 Operating instruction*. Dostupné také z: http://www.christiani.de/pdf/54030_operating_instructions_EN.pdf

SIEMENS[8], 2018. HMI Getting Started Manual. *Manual*. Dostupné také z: https://cache.industry.siemens.com/dl/files/596/73505596/att_77660/v1/GettingStarted_en-US.pdf

SIEMENS[9], 2018. *HMI TP700 Comfort Brochure*. Dostupné také z: https://w5.siemens.com/italy/web/AD/ProdottieSoluzioni/Sistemiautomazionenew/Eventi/Documents/presentazioni%20simatic%20live/SIMATIC%20HMI_comfort%20panels.pdf?istablet=true

SUNON, 2018. *Mag Lev Motor Fan*. Dostupné také z: https://www.jameco.com/Jameco/Products/ProdDS/695050-Revised.pdf

VÍTEČEK, Antonín a Miluše VÍTEČKOVÁ, 2013. *Closed-loop control of mechatronic systems*. Ed. 1st. Ostrava: VŠB - Technical University of Ostrava. ISBN isbn978-80-248-3149-7.

WAGNEROVA, Renata, 2019. Automation. *System Identification*. Dostupné také z: http://fs1.vsb.cz/352/7_WAGNEROVA/Automation/Identification_ang.pdf

WEI, Fanjie, 2016. The PLC-based Industrial Temperature control system: Design and implementation. In: *MATEC web of conferences*. s. 6. DOI: 10.1051/matecconf/201710003031. Dostupné také z: https://www.matec-conferences.org/articles/matecconf/pdf/2017/14/matecconf_gcmm2017_03031.pdf

ACKNOWLEDGEMENT

First, and most of all, I would like to thank my supervisor Ing. Miroslav Mahdal, Ph.D. for his expertise, assistance, guidance and patience throughout the process of this thesis. Without your help this thesis would not have been possible.

I would also like to express my gratitude and thanks to the exceptional faculty of the control system and instrumentation department.

This project has been supported by SP2019 / 51 Applied Research in the Area of Machine and Process Control supported by the Ministry of Education, Youth and Sports.

Last of all, I would like to thank my friends and everyone else who helped me contribute to this project.

APPENDIX

CD content

- Thesis document in PDF.
- TIA Portal working file.
- PCB design files.
- MATLAB files.